facebook

Artificial Intelligence Research

Object Recognition

Georgia Gkioxari

Content

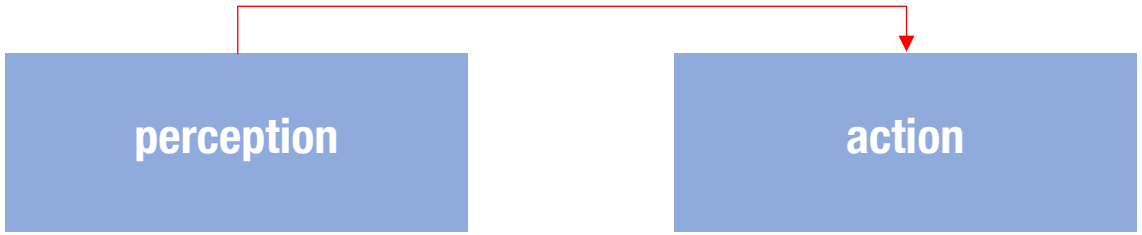
- A Brief Intro
- Visual Recognition: Problem Definitions & Tasks
  - Semantic Segmentation
    - FCN
  - Object Detection
    - Fast(er) R-CNN
  - Instance Segmentation & Pose Prediction
    - Mask R-CNN
- ImageNet in 1H: Important Lessons
- Useful Tips

Content

- A Brief Intro
- Visual Recognition: Problem Definitions & Tasks
  - Semantic Segmentation
    - FCN
  - Object Detection
    - Fast(er) R-CNN
  - Instance Segmentation & Pose Prediction
    - Mask R-CNN
- ImageNet in 1H: Important Lessons
- Useful Tips

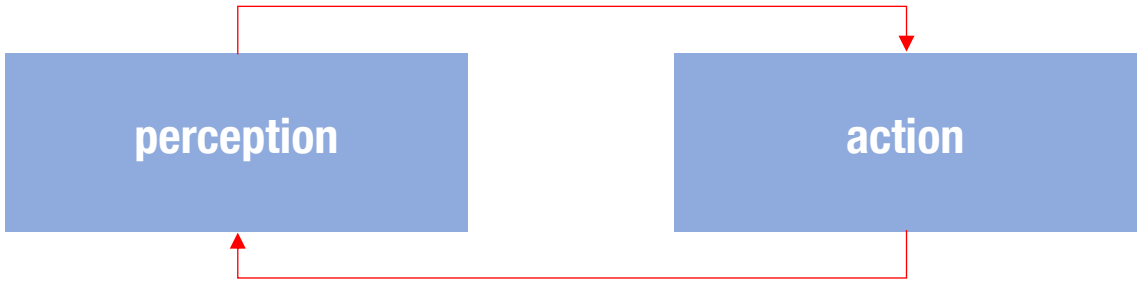
Why Perception?

Why Perception?



Find food
Avoid predator
....

Why Perception?



Why Perception?

A Brief Intro

A Brief Intro

Invariance: Transformation changes have no effect

$$f(g(x)) = f(x), \forall x$$

Equivariance: Transformation changes in input lead to corresponding changes in output

$$\exists M_g, f(g(x)) = M_g(f(x)), \forall x$$

A Brief Intro

Classification desires invariant representations: output a label

Detection/Segmentation desires equivariant representations:

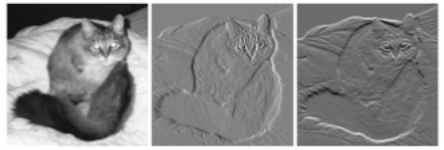
- Translated Object → Translated Mask
- Scaled Object → Scaled Mask

A Brief Intro

- Convolutions are translation-equivariant
- Fully ConvNets (FCN) are translation-equivariant
- ConvNets become translation-invariant due to fully connected layers or global pooling layers

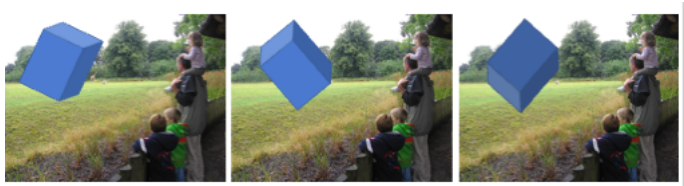
A Brief Intro

Sources of Variation



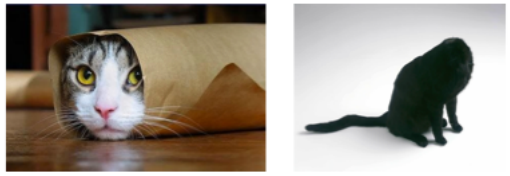
Photometric

Translation

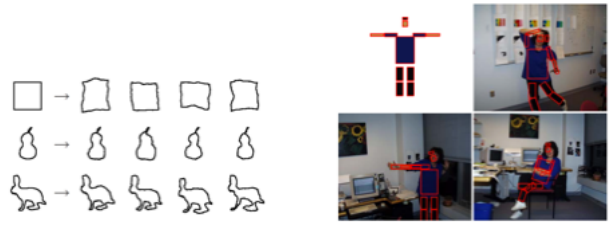


Rotation, rigid pose

Scale



Occlusion (self, other)



Deformation (local, articulated)

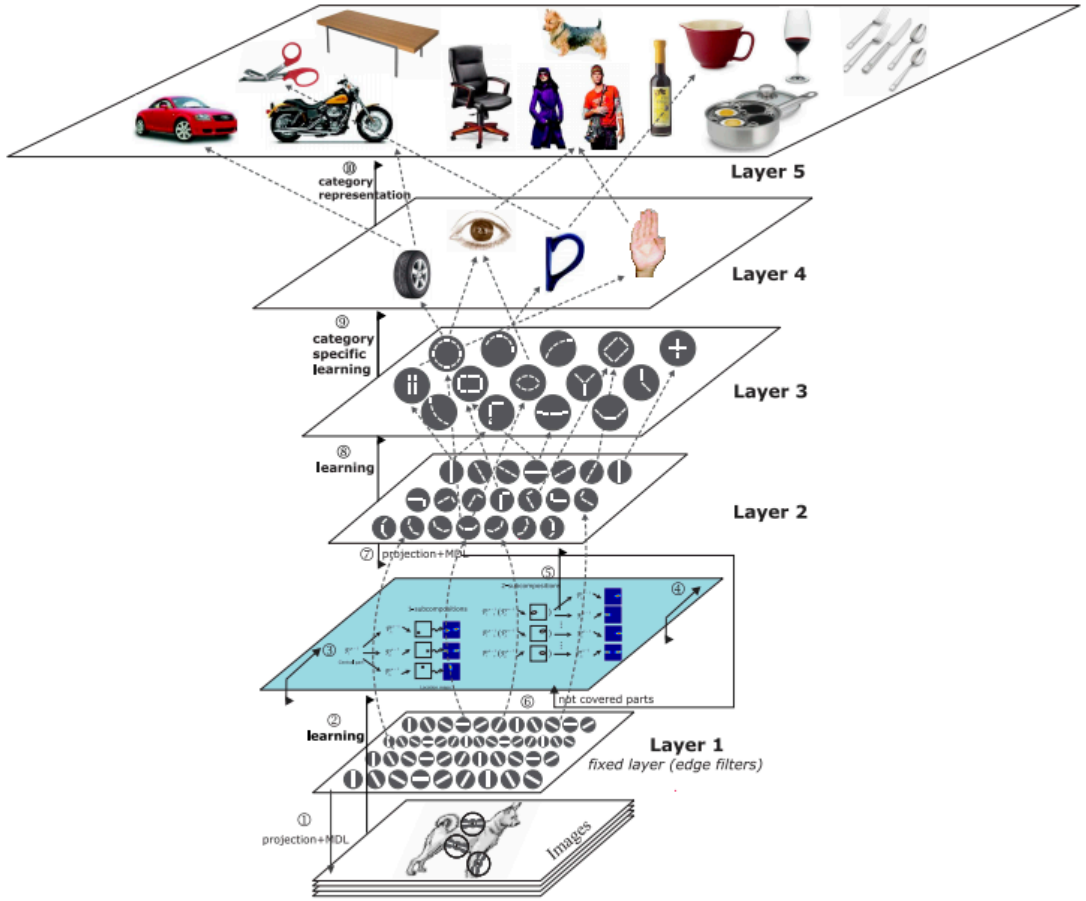
Intra-class
(appearance, structural)

Context
(clutter, spurious corr.)

A Brief Intro

Recognition is the science and engineering of models that have effective invariance and equivariance properties

A Brief Intro



- Scenes are composed of “stuff” and “things” (objects)
- Object are composed of parts
- Parts are composed of subparts
- Subparts are built from edges

Learning Hierarchical Representations of Object Categories for Robot Vision. Leonardis & Fidler. 2007.

A Brief Intro

When designing a neural network, think how

- You are modeling equivariance/invariance

A Brief Intro

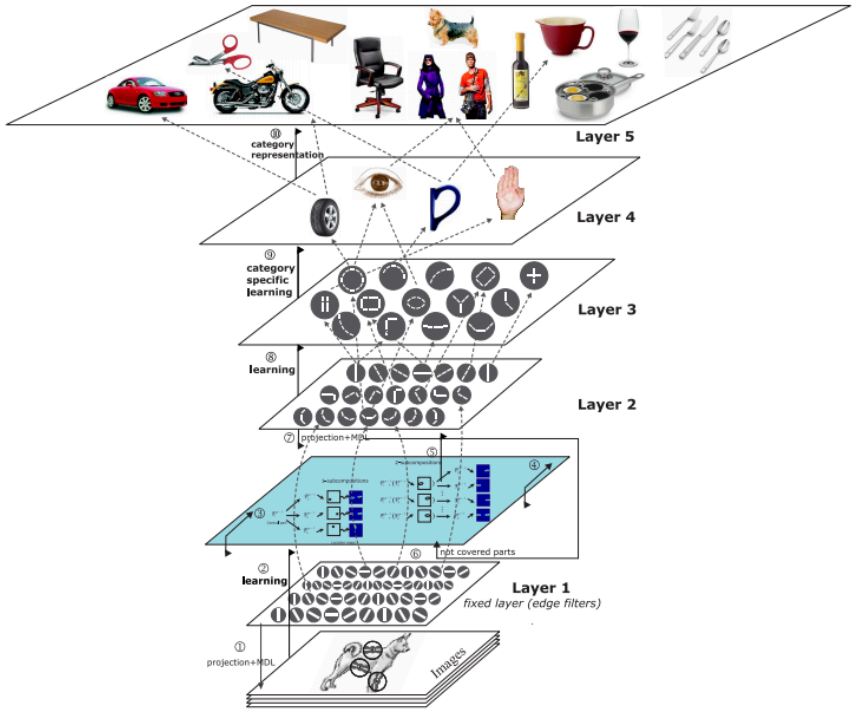
When designing a neural network, think how

- You are modeling equivariance/invariance
- You are leveraging the compositional structure

A Brief Intro

When designing a neural network, think how

- You are modeling equivariance/invariance
- You are leveraging the compositional structure



A Brief Intro

When designing a neural network, think how

- You are modeling equivariance/invariance
- You are leveraging the compositional structure
- You are modeling the computational task as an algorithm

David Marr, Vision: A Computational Investigation Into the Human Representation and Processing of Visual Information

A Brief Intro

When designing a neural network, think how

- You are modeling equivariance/invariance
- You are leveraging the compositional structure
- You are modeling the computational task as an algorithm
- You are organizing the computations so that the solution is efficient

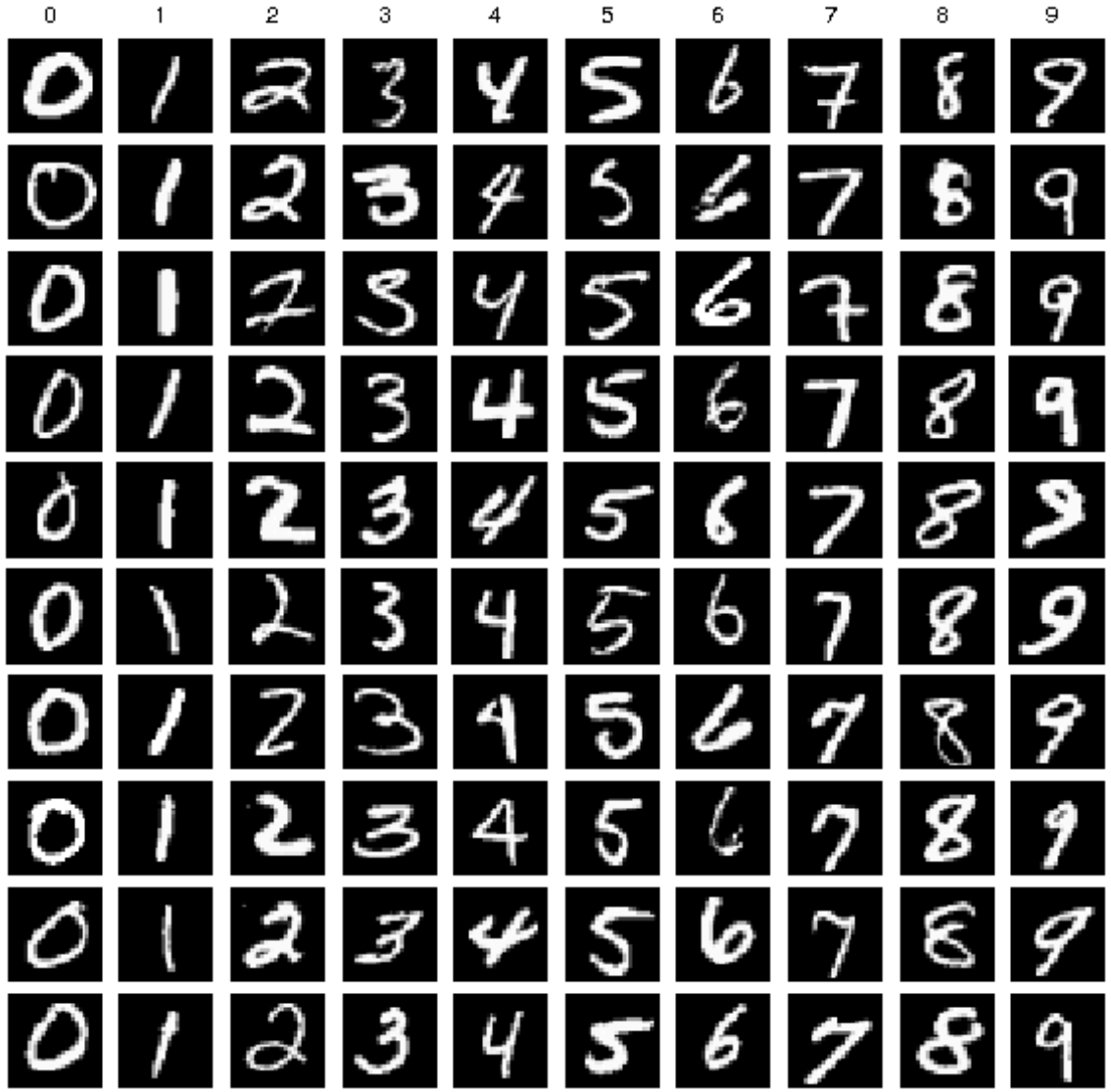
A Brief Intro

When designing a neural network, think how

- You are modeling equivariance/invariance
- You are leveraging the compositional structure
- You are modeling the computational task as an algorithm
- You are organizing the computations so that the solution is efficient

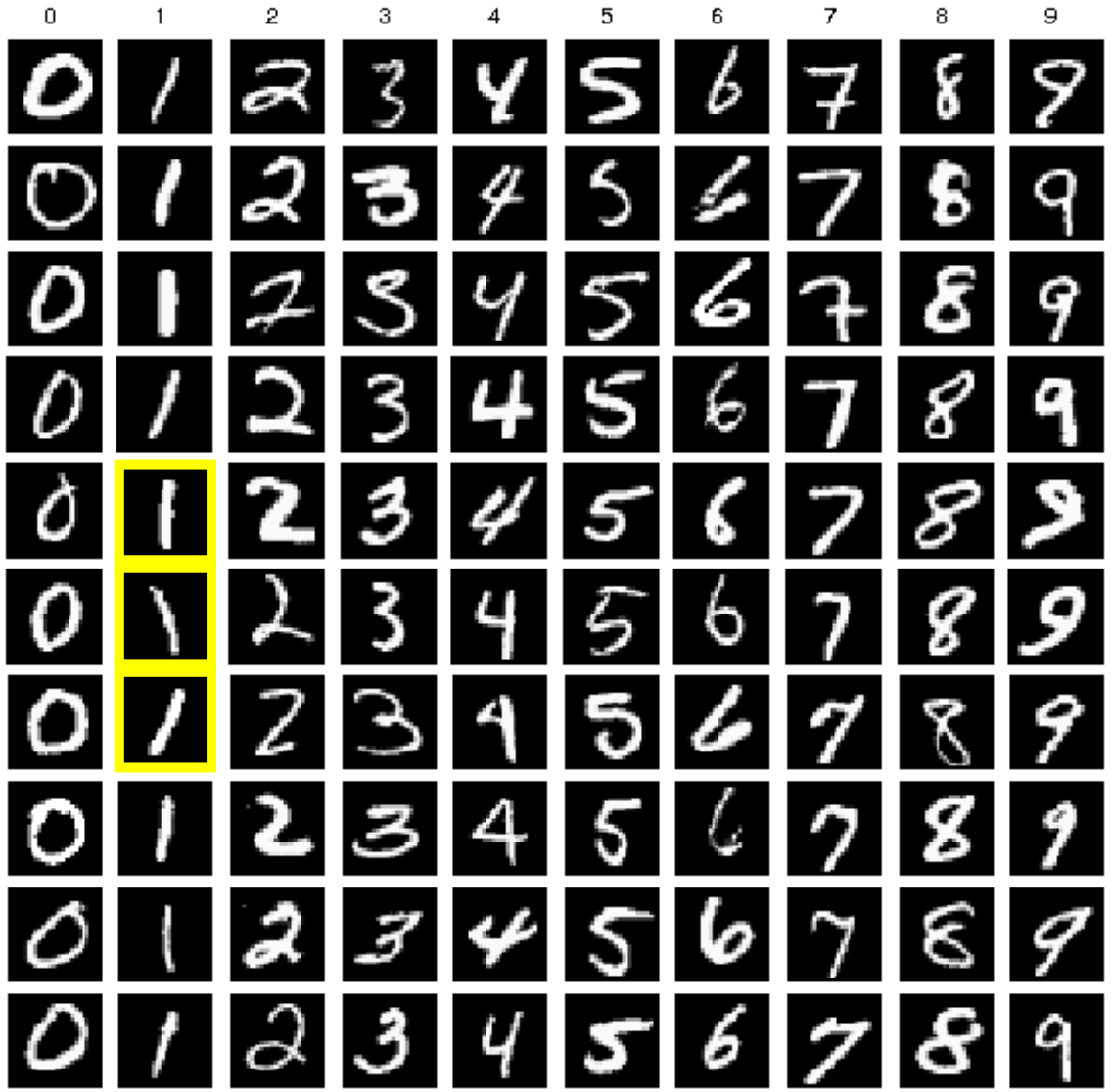
→ **Success**

Isolated Handwritten Digit Recognition



Task: classify into categories {0, 1, …, 9}

Isolated Handwritten Digit Recognition

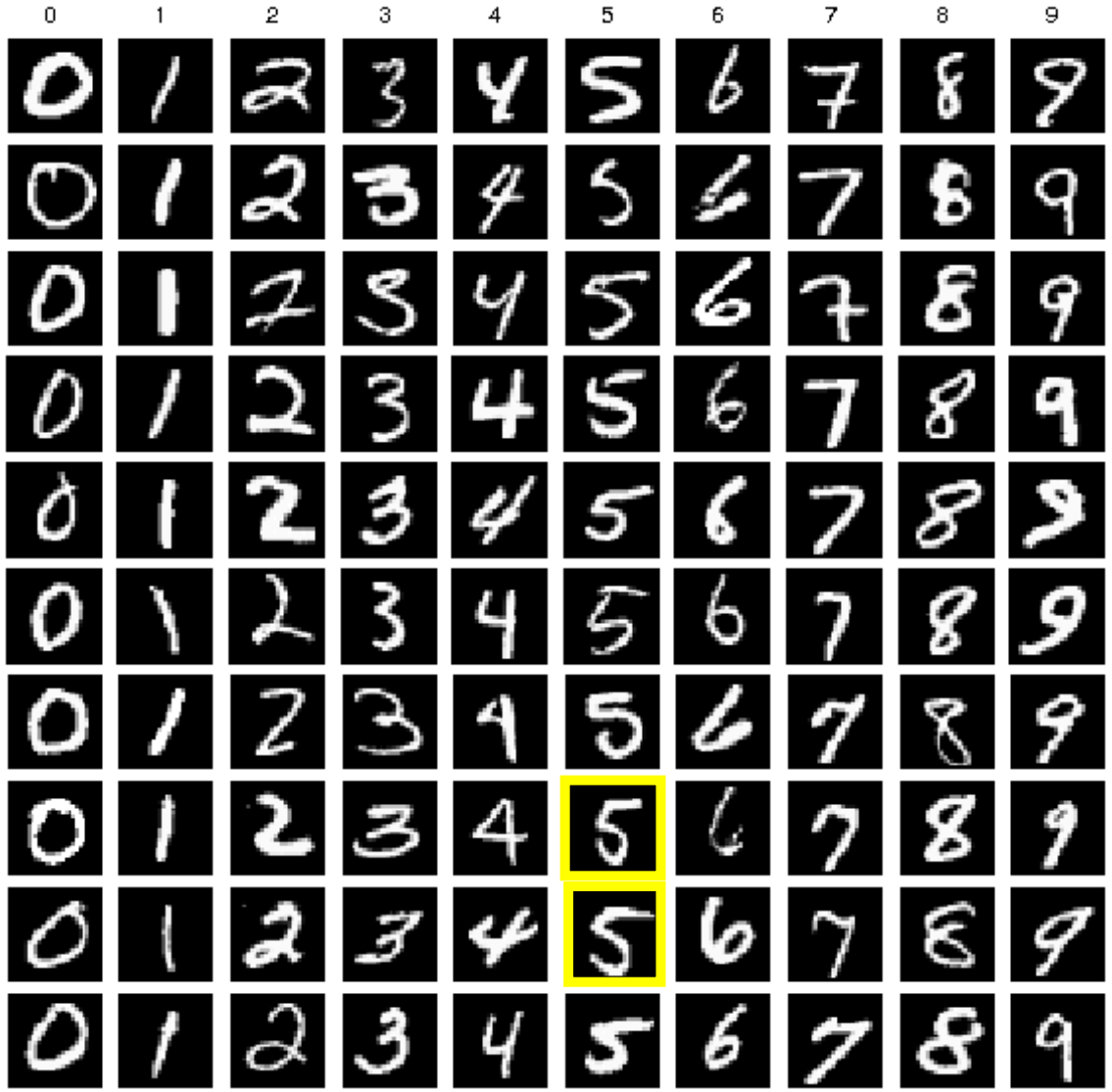


Task: classify into categories {0, 1, …, 9}

Sources of variation:

- Small rotations (in plane)

Isolated Handwritten Digit Recognition

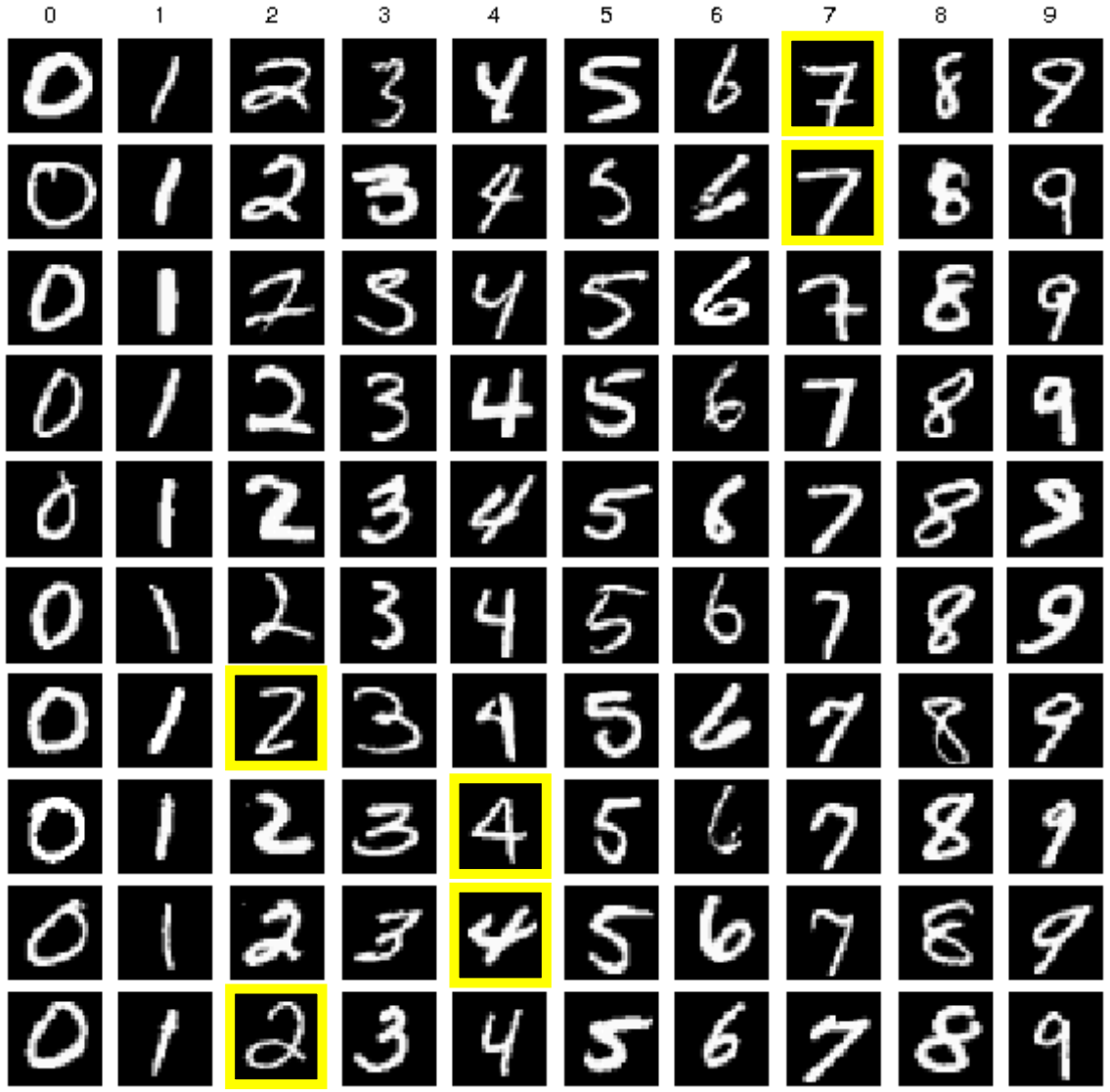


Task: classify into categories {0, 1, …, 9}

Sources of variation:

- Small rotations (in plane)
- Local deformation

Isolated Handwritten Digit Recognition

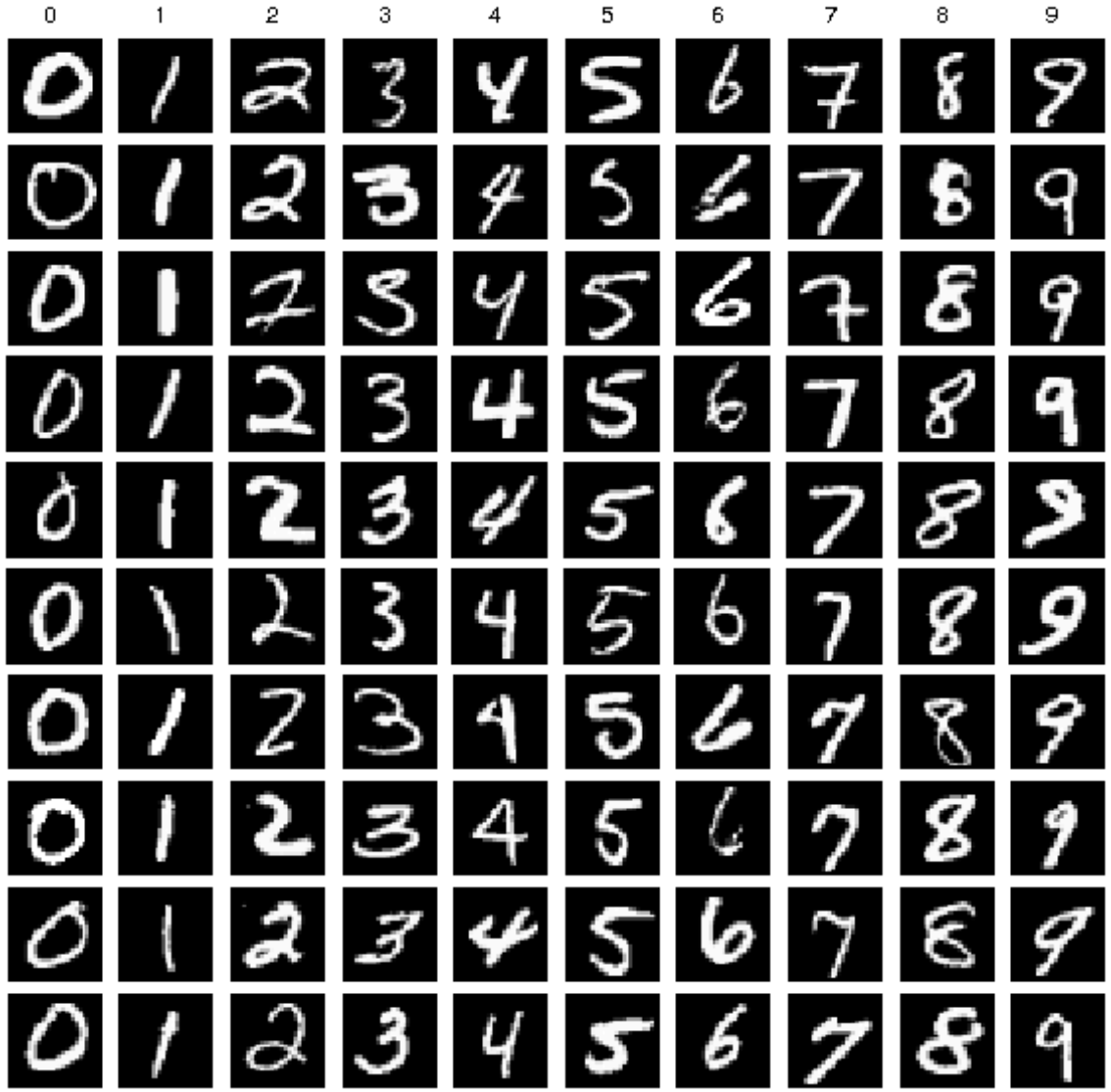


Task: classify into categories {0, 1, …, 9}

Sources of variation:

- Small rotations (in plane)
- Local deformation
- Intra-class variation (style; see 4s and 7s)

Isolated Handwritten Digit Recognition



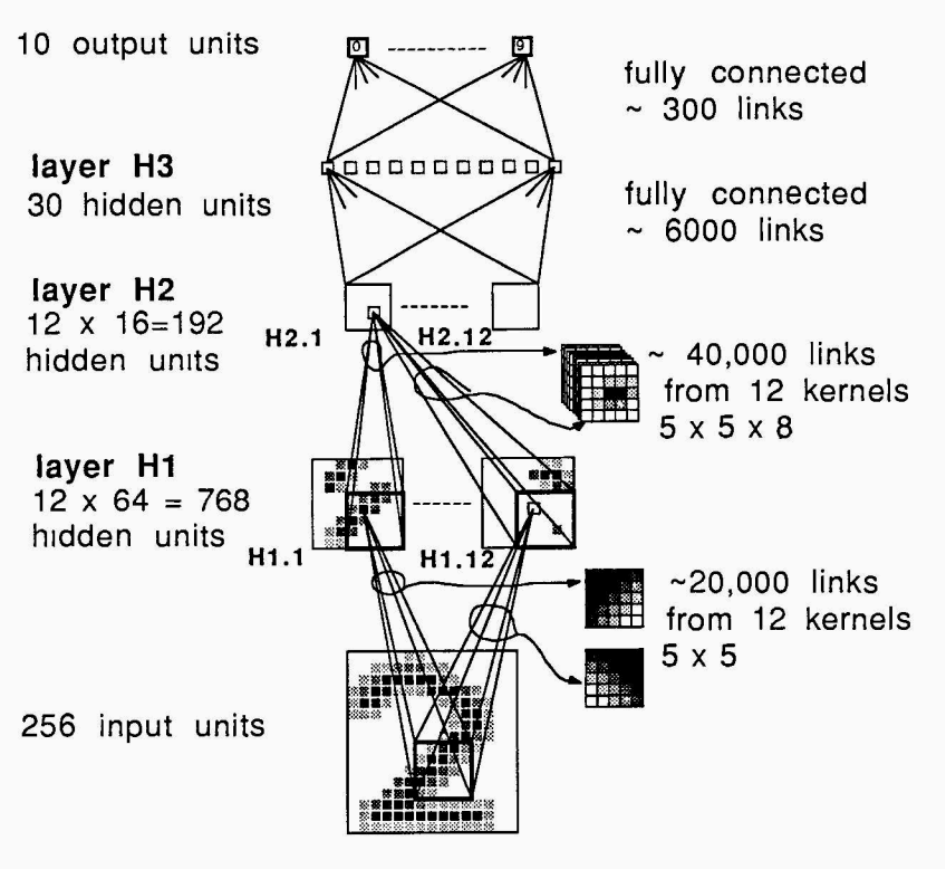
Task: classify into categories {0, 1, …, 9}
Sources of variation:

- Small rotations (in plane)
- Local deformation
- Intra-class variation (style; see 4s and 7s)

Non-sources of variation:

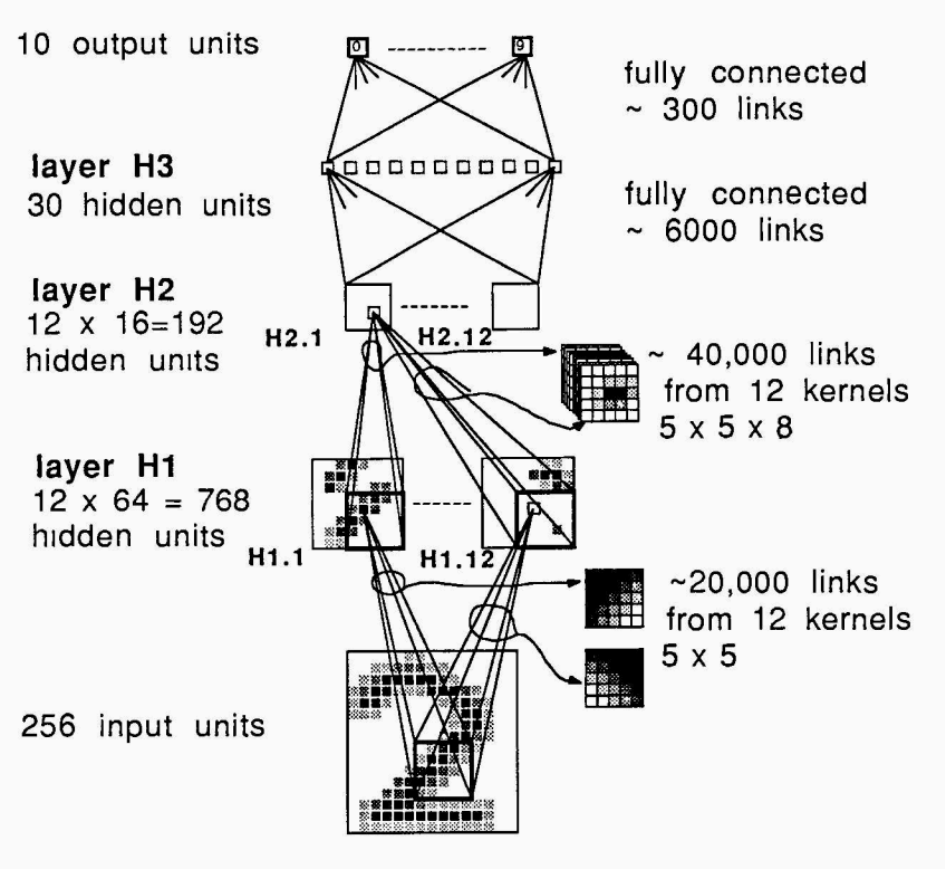
- Photometric
- Translation
- Rotation (out of plane)
- Scale
- Occlusion
- Articulated deformation
- Context

A Brief Intro: Digit Classification



Backpropagation Applied to Handwritten Zip Code Recognition. LeCun et al. 1989.

A Brief Intro: Digit Classification



- Two convolutional layers
- One fully connected layer
- Spatial contraction via strided convolution (subsampling)
- Training via SGD with backpropagation

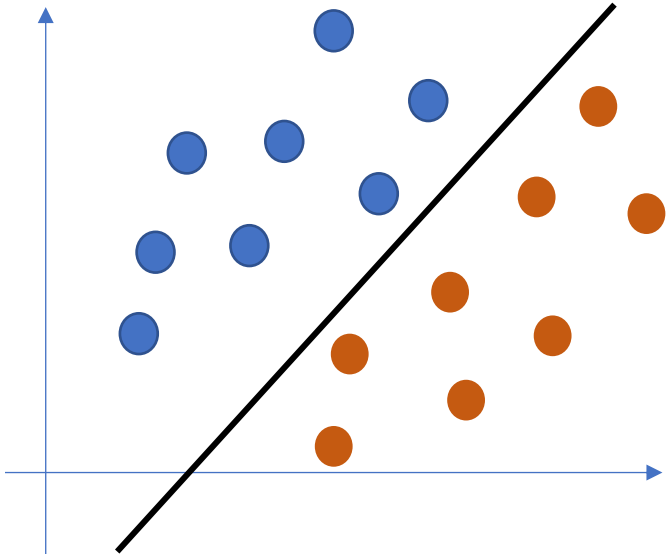
Backpropagation Applied to Handwritten Zip Code Recognition. LeCun et al. 1989.

A Brief Intro: Optimization

Assume a dataset $D = \{(x_i, y_i)\}_{i=0}^{N-1}$

We want to learn a function f such that $y_i = f(x_i), \forall i$

E.g. binary classification $y \in \{-1, 1\}$ and $x \in \mathbb{R}^2$



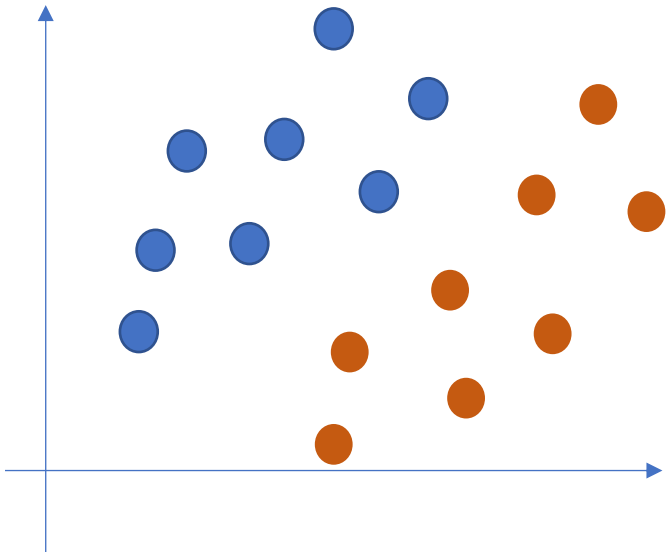
Linearly separable so f is a linear function of x

A Brief Intro: Optimization

Assume a dataset $D = \{(x_i, y_i)\}_{i=0}^{N-1}$

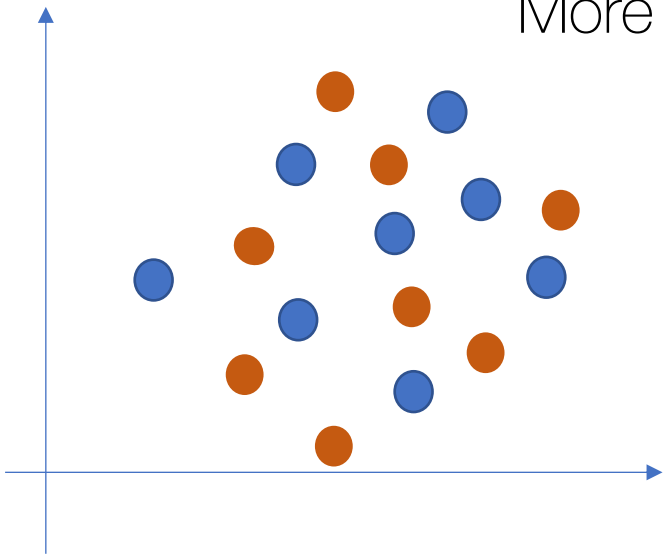
We want to learn a function f such that $y_i = f(x_i), \forall i$

E.g. binary classification $y \in \{-1, 1\}$ and $x \in \mathbb{R}^2$



This never happens!

More common!



A Brief Intro: Optimization

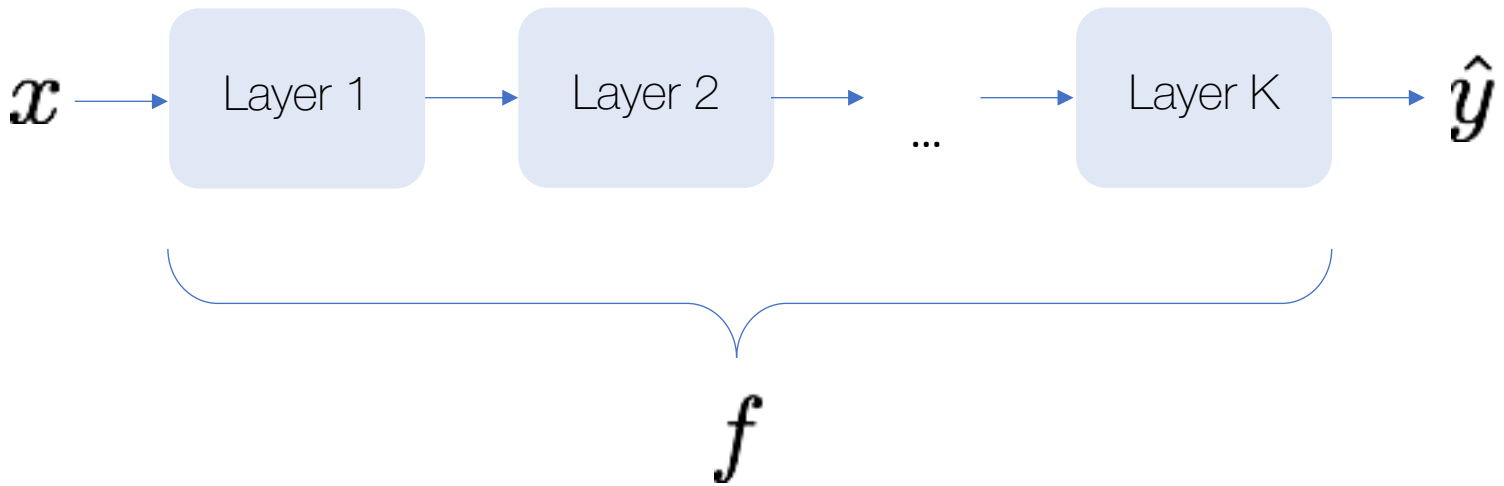
Assume a dataset $D = \{(x_i, y_i)\}_{i=0}^{N-1}$

We want to learn a function f such that $y_i = f(x_i), \forall i$

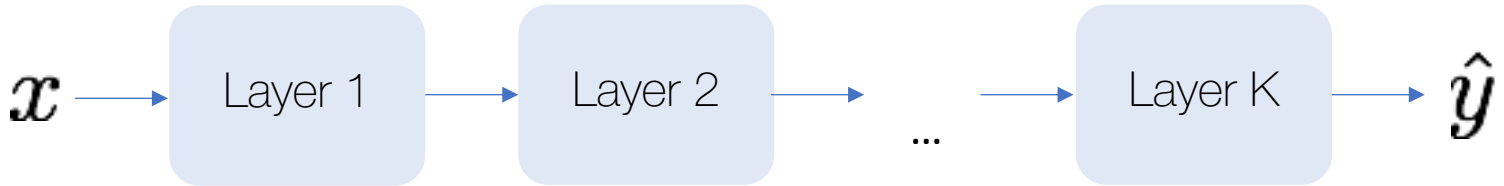
Deep Learning

Find a non-linear function approximator f that best describes our dataset D

A Brief Intro to Deep Learning



A Brief Intro to Deep Learning

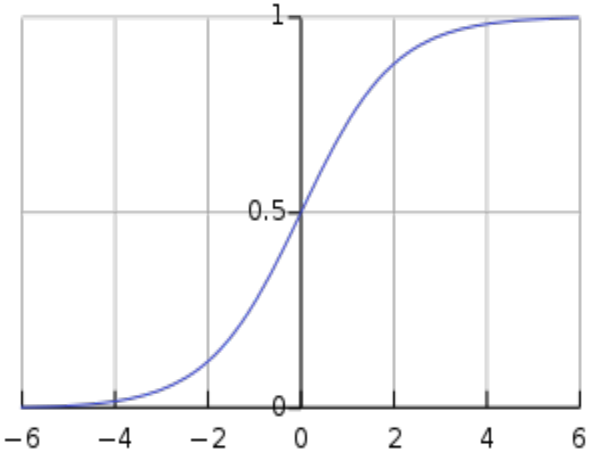


Learnable parameters w_j

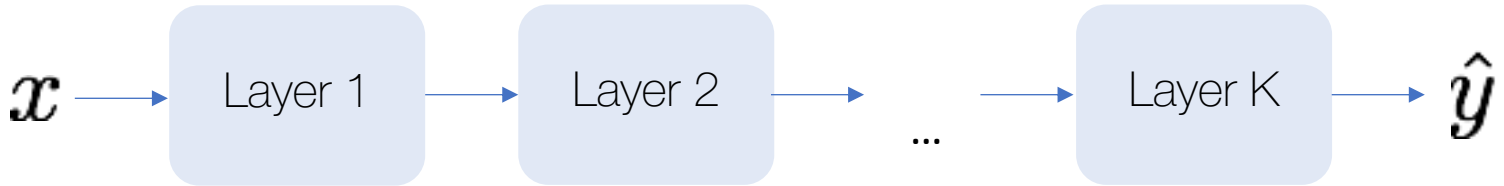
Layer j performs a specific operation, e.g.

$$out_j = \sigma(w_j \cdot in_j)$$

Activation functions σ insert non-linearities, e.g.

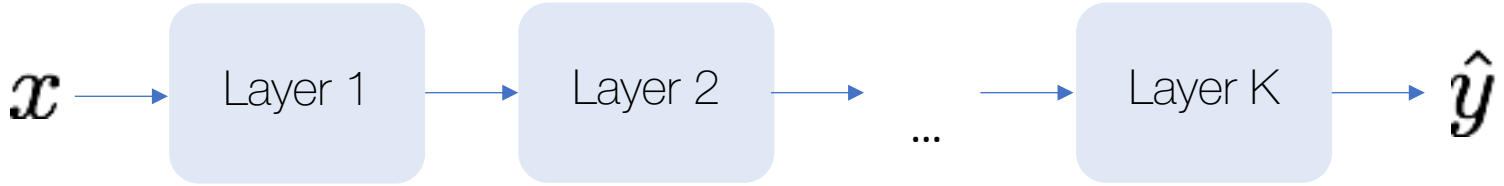


A Brief Intro to Deep Learning



We want to find $\{w_1, w_2, ..., w_K\}$ s.t. $\hat{y}_i \approx y_i$

A Brief Intro to Deep Learning



We want to find $\{w_1, w_2, ..., w_K\}$ s.t. $\hat{y}_i \approx y_i$

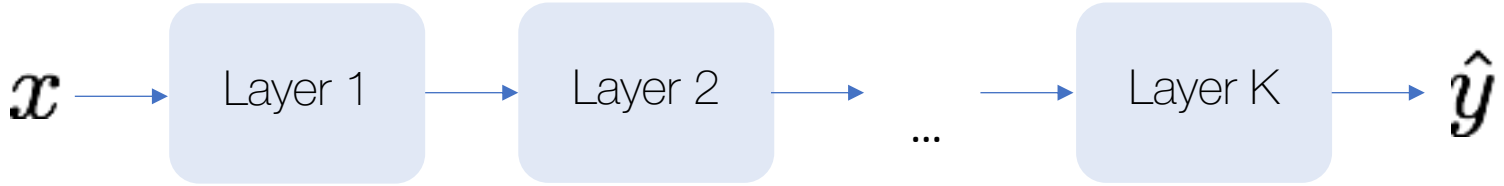
Loss functions

Quantify the error between prediction and ground truth

e.g.

$$l = \frac{1}{N} \sum_{i=0}^{N-1} (\hat{y}_i - y_i)^2$$

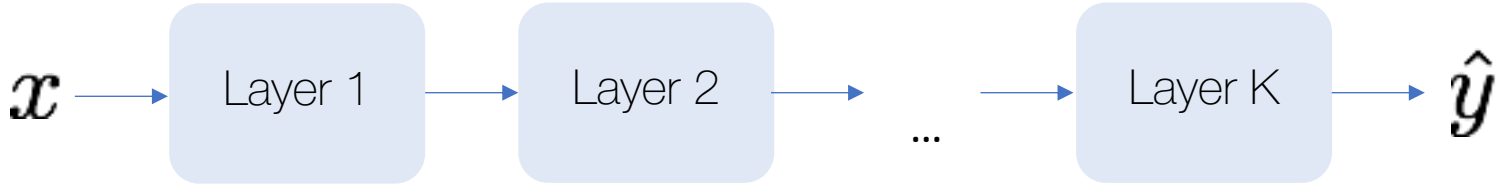
A Brief Intro to Deep Learning



Optimization

$$\min_{\{w_1, w_2, \ldots, w_K\}} l = \min_{\{w_1, w_2, \ldots, w_K\}} \frac{1}{N} \sum_{i=0}^{N-1} (\hat{y}_i - y_i)^2$$

A Brief Intro to Deep Learning



Optimization: Gradient Descent

At t=0, $w_j^{(0)} = 0$

At t-th iteration:

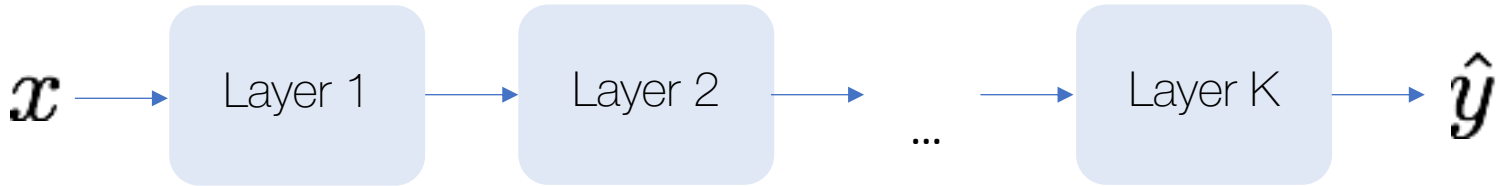
$$w_j^{(t)} = w_j^{(t-1)} - \eta \frac{dl}{dw_j}$$

Learning rate

Gradient

A Brief Intro to Deep Learning

Remember that
$x = in_1$
$\hat{y} = out_K$
$out_j = in_{j+1}$



Gradient Computation: Chain Rule

$$\frac{dl}{dw_j} = \frac{dl}{d\hat{y}} \cdot \frac{dout_K}{din_K} \cdot \frac{dout_{K-1}}{din_{K-1}} \cdot \ldots \cdot \frac{dout_j}{dw_j}$$

Visual Recognition Tasks

Digit recognition

Image Classific.

Semantic Segmentation

Object Detection

Instance Segmentation

Increasing complexity

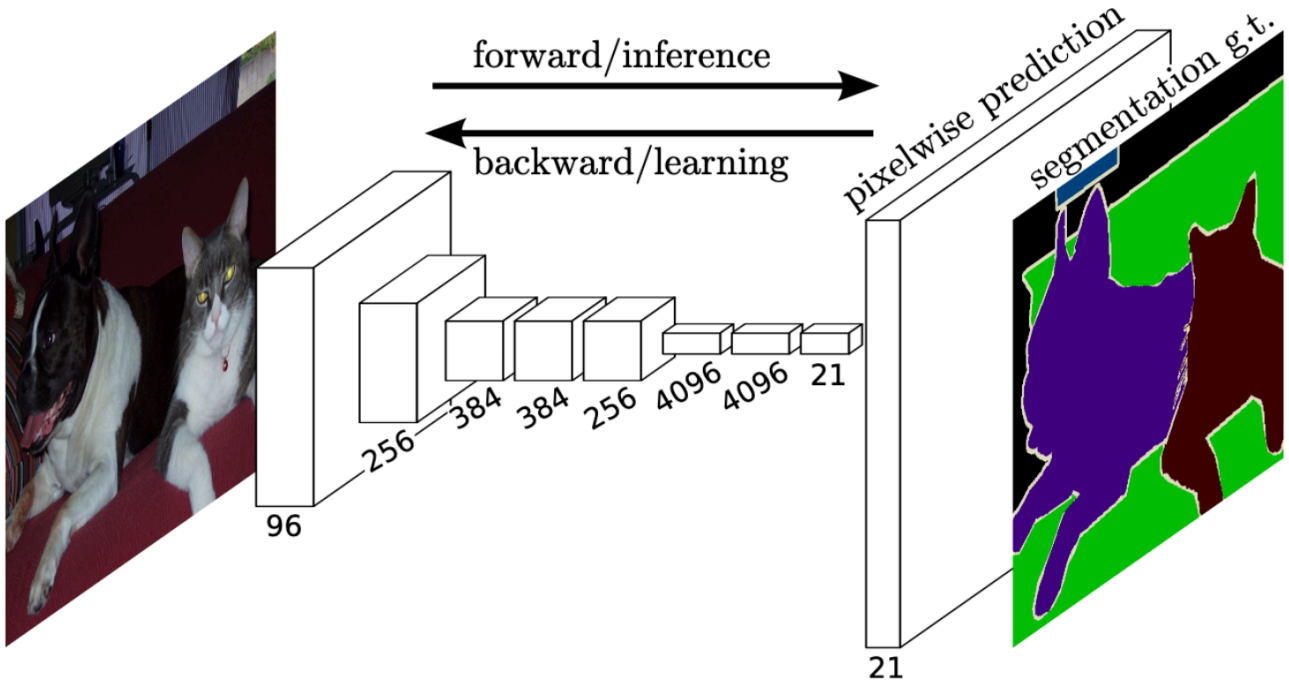
Content

- A Brief Intro
- Visual Recognition: Problem Definitions & Tasks
  - Semantic Segmentation
    - FCN
  - Object Detection
    - Fast(er) R-CNN
  - Instance Segmentation & Pose Prediction
    - Mask R-CNN
- ImageNet in 1H: Important Lessons
- Useful Tips

Semantic Segmentation

- Given an input image I, mark each pixel in I with a class label
- Unlike image classification, predictions are cast for every pixel
- Metrics:
  - Mean Accuracy
  - Mean Intersection over Union

FCN



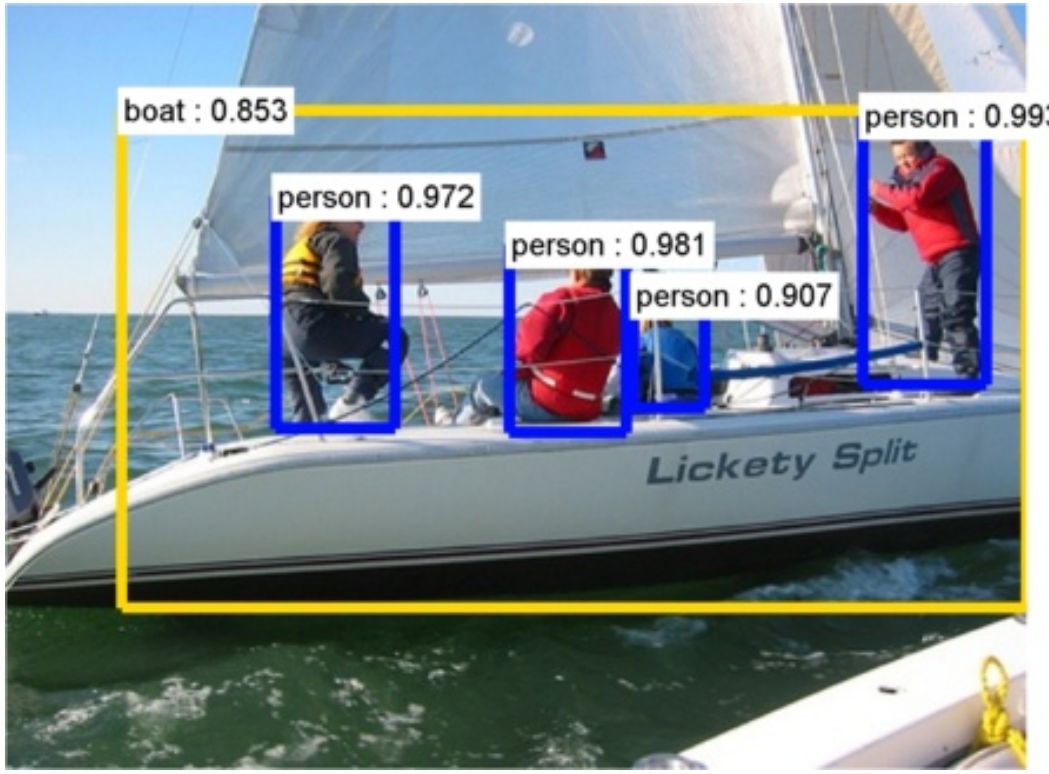
- Input image I of size H x W x 3
- Output logits of size H x W x C, where C is the number of object categories in the dataset.

Long et al, FCN for Semantic Segmentation, CVPR 2015

FCN: Losses

- Assume J $\in \mathbb{R}^{HxWxC}$ are the logits predicted by FCN & $J_{gt} \in [0, C-1]^{HxW}$ is the ground truth
- Binary classification vs. Softmax

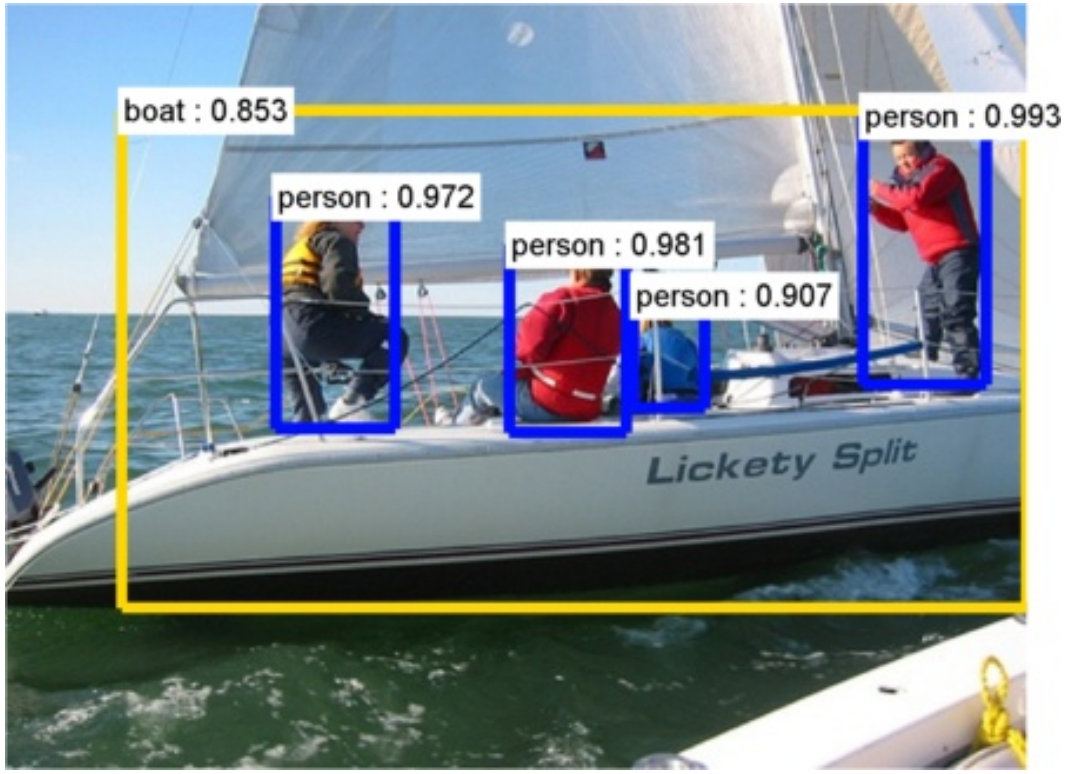
Object Detection



What

Where

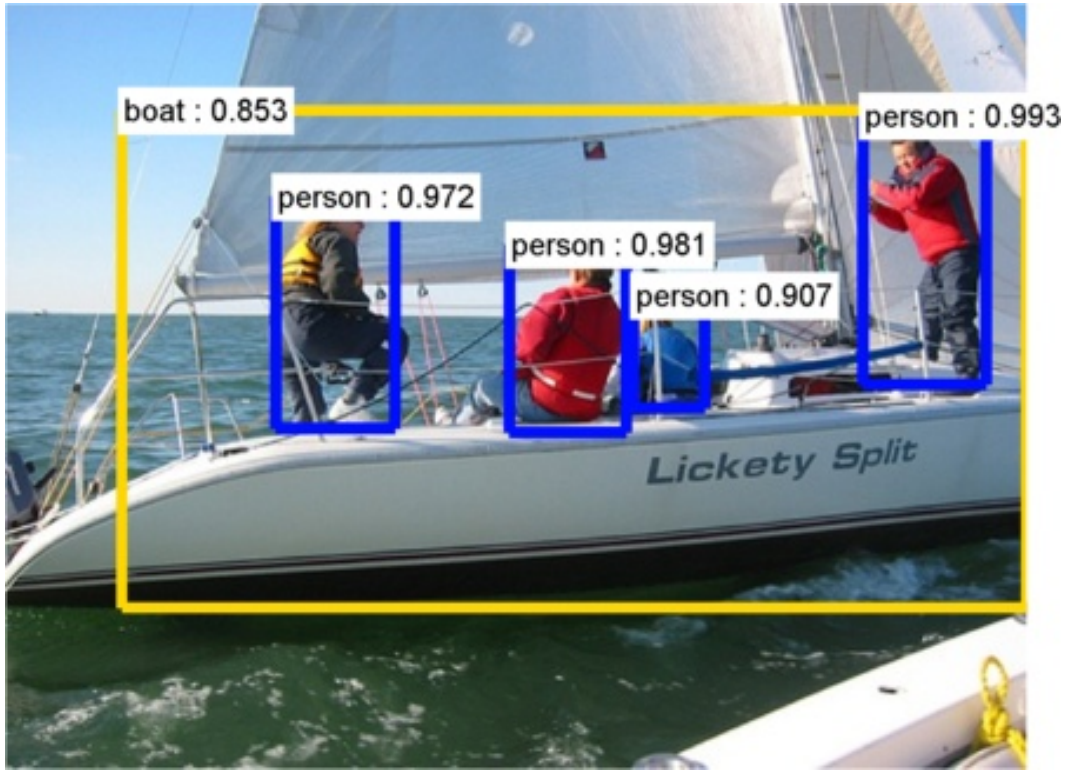
Object Detection



Object Detection

Image Classification

Object Detection



Task

- Assume C object classes
- Localize and classify all objects in an image

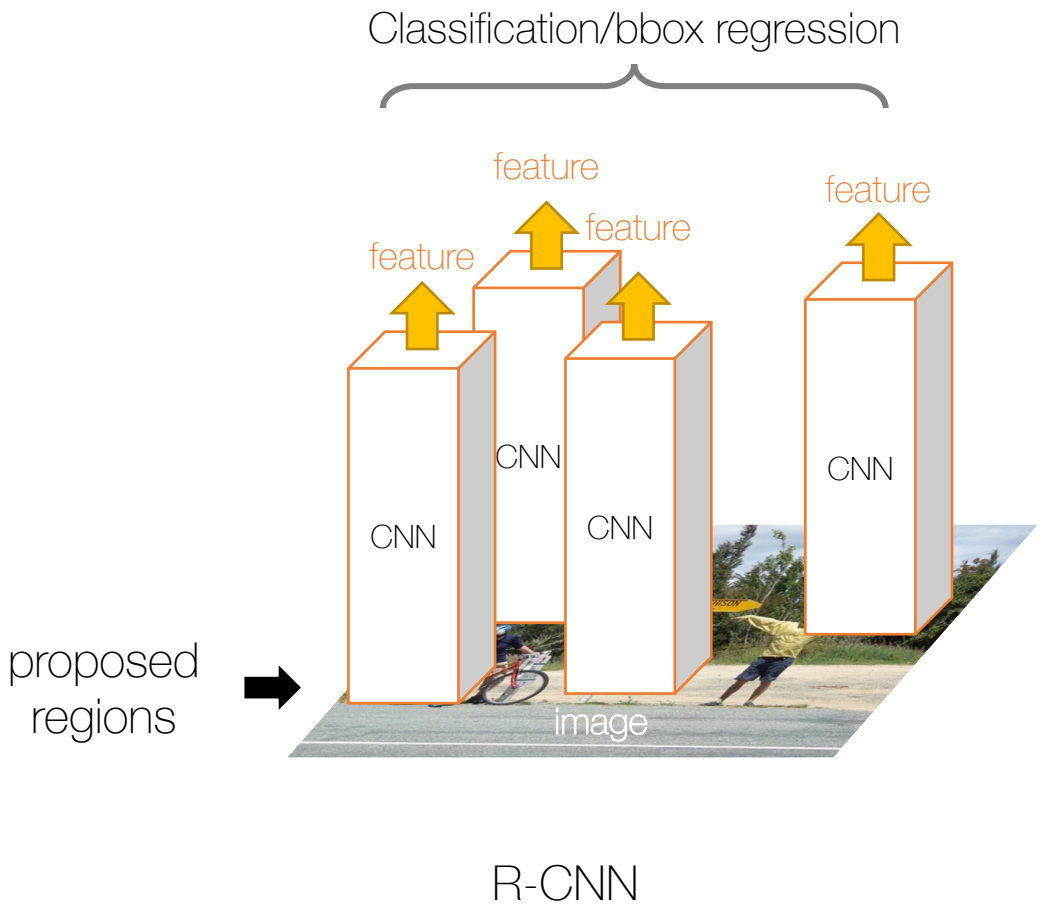
True Positives

- A detection b is a true positive if
  ov(b, gt) > 0.5 and class(b) = class(gt)

False Positives

- Mislocalized and misclassified detections
- Duplicate detections are penalized

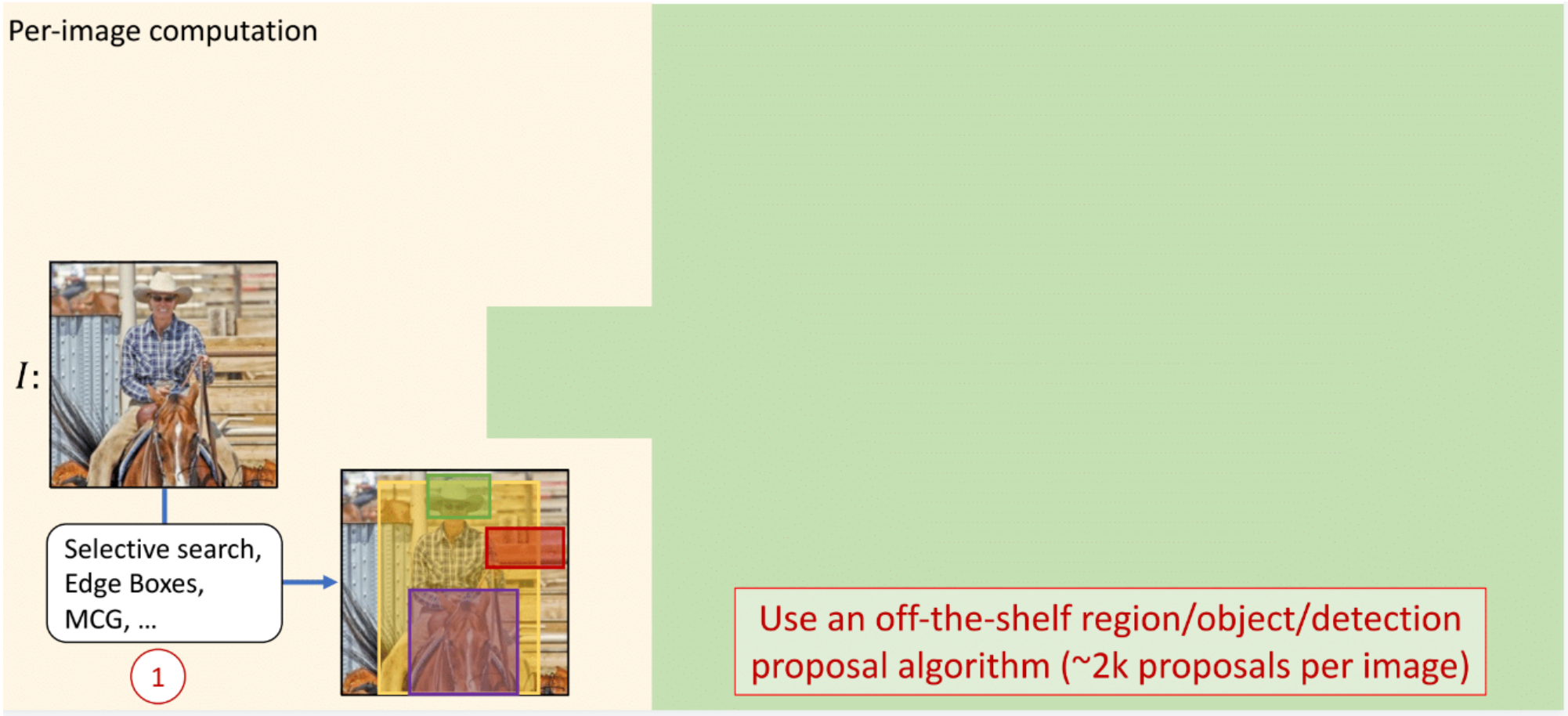
Object Detection Frameworks: R-CNN



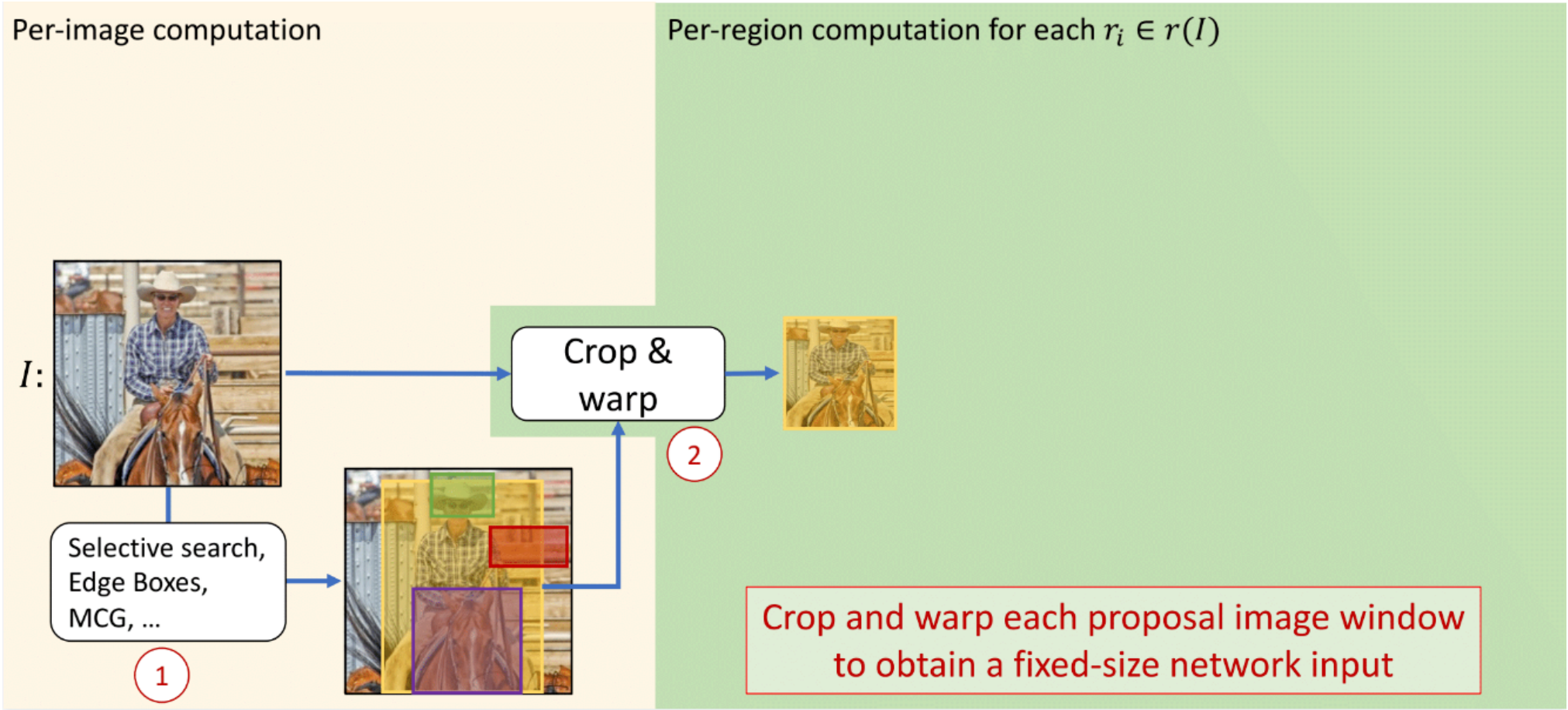
R-CNN

Girshick et al, CVPR 2014

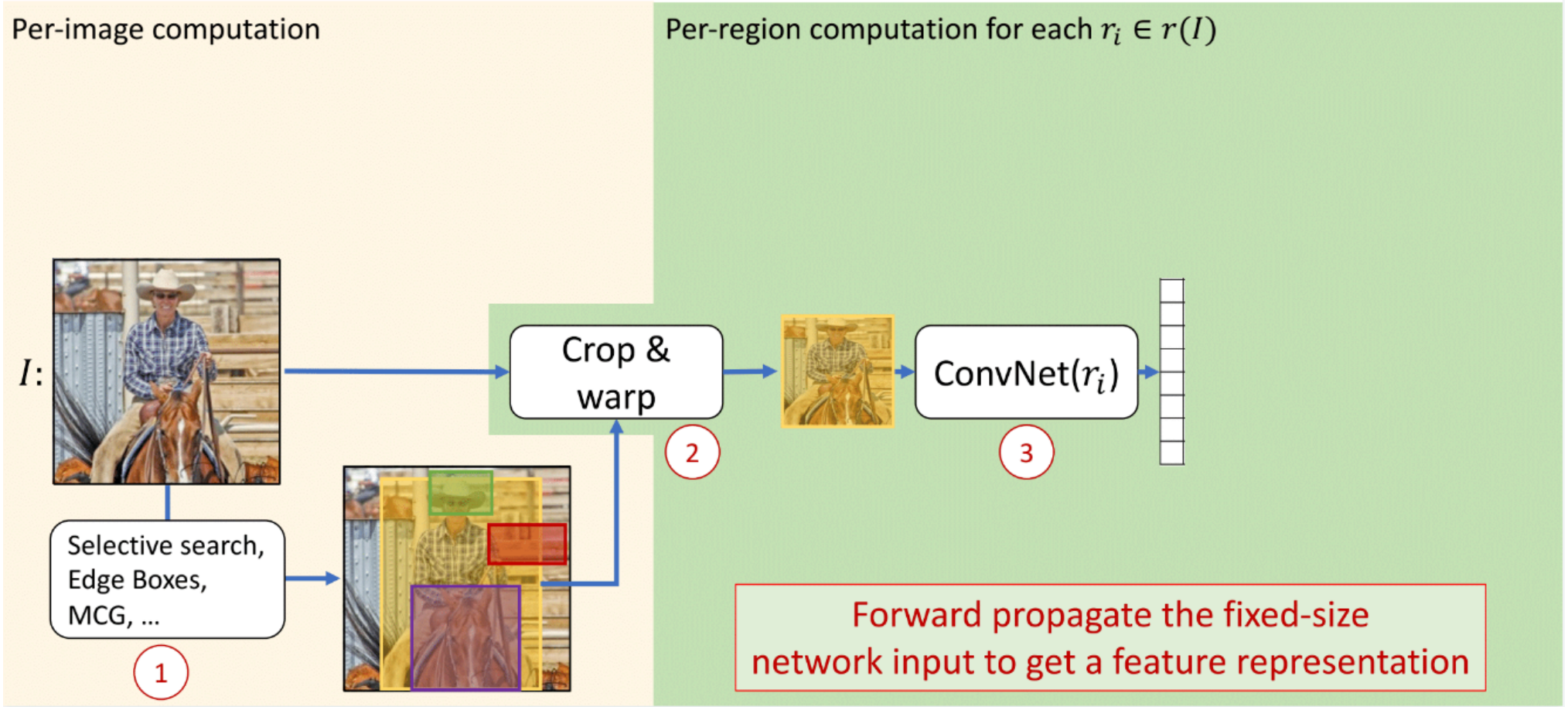
Object Detection Frameworks: R-CNN



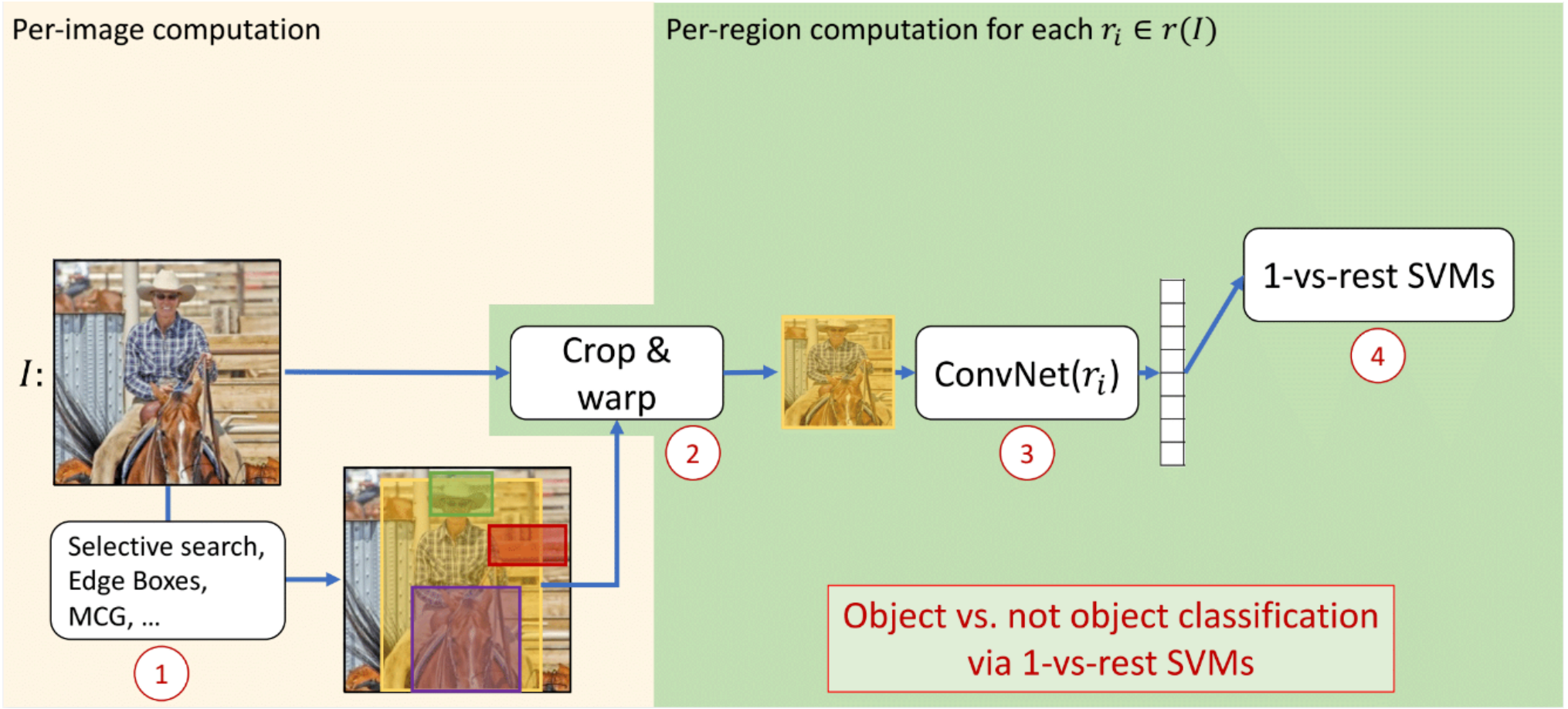
Object Detection Frameworks: R-CNN



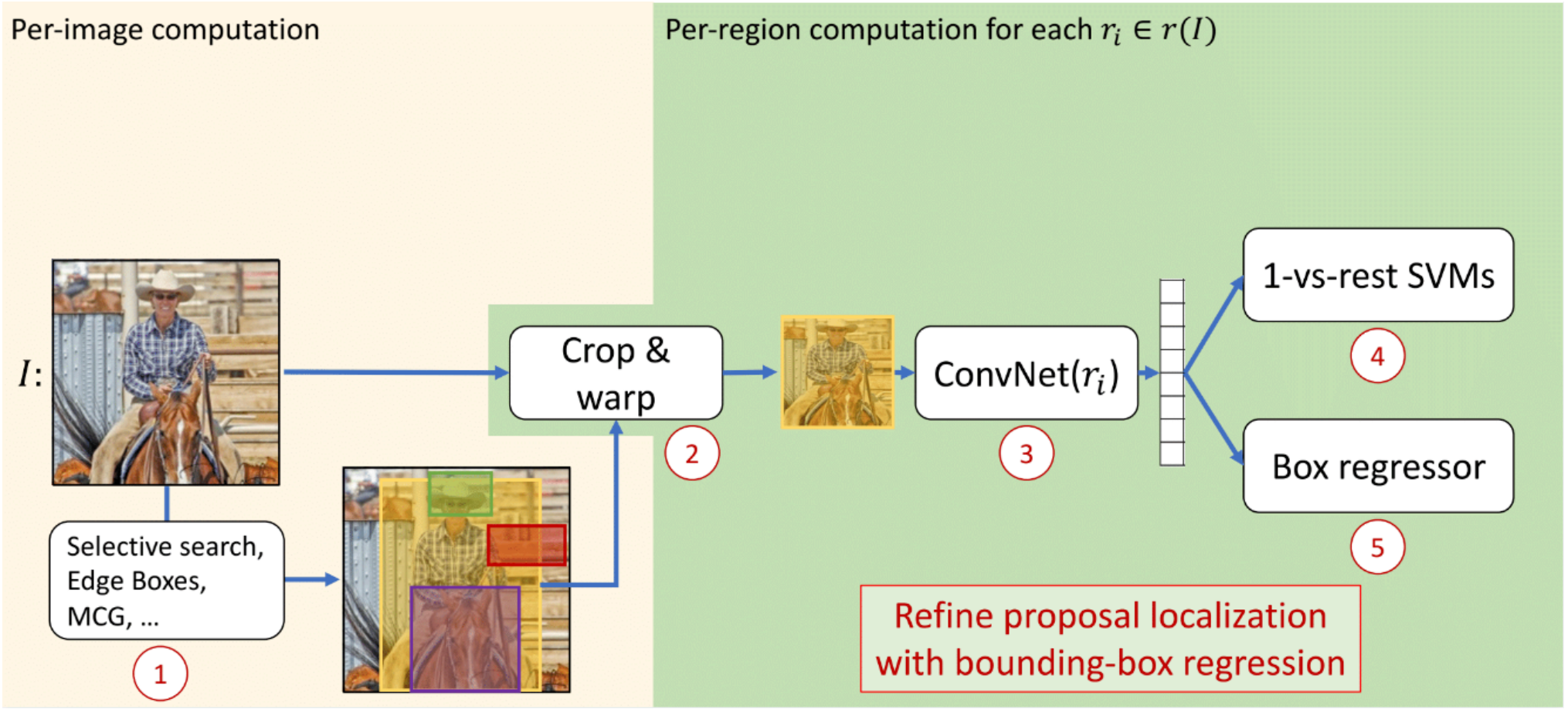
Object Detection Frameworks: R-CNN



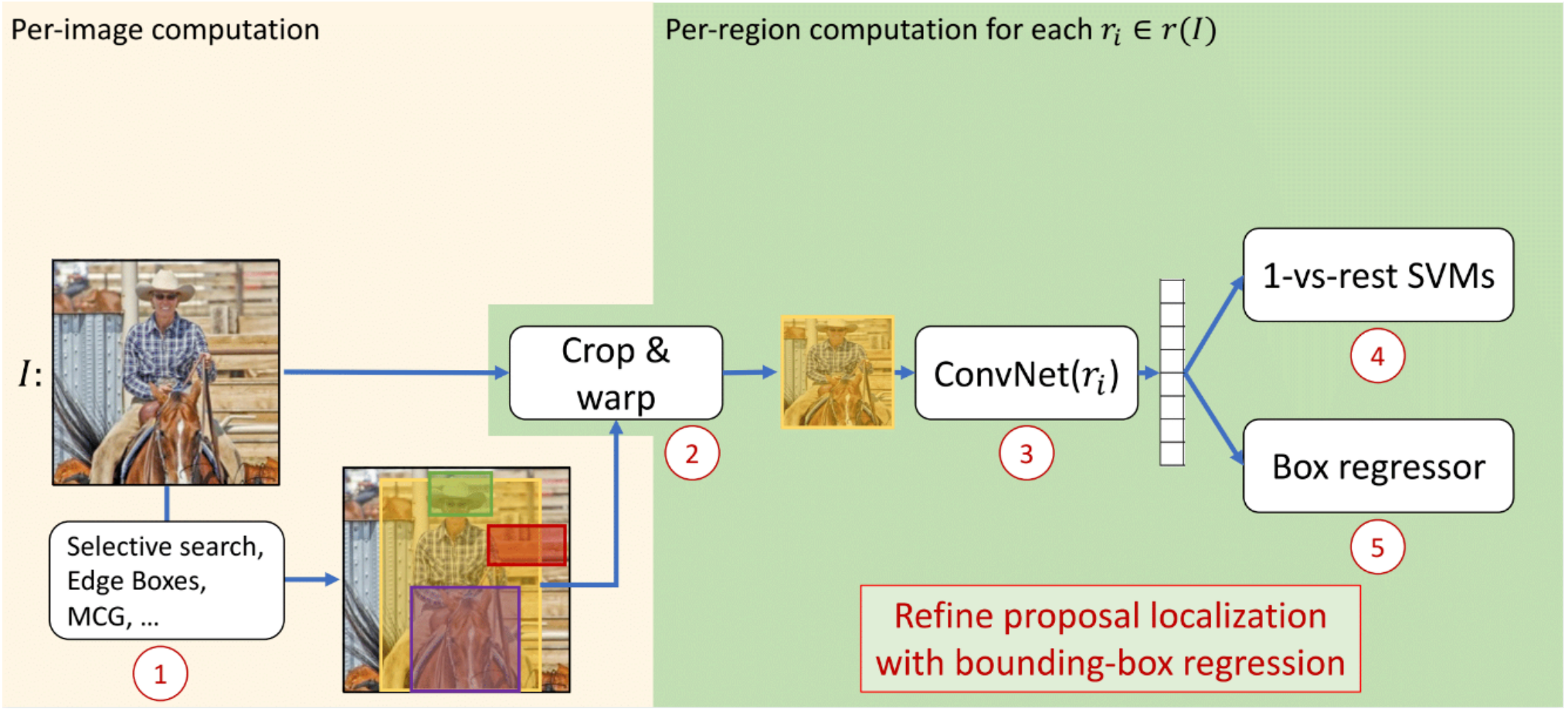
Object Detection Frameworks: R-CNN



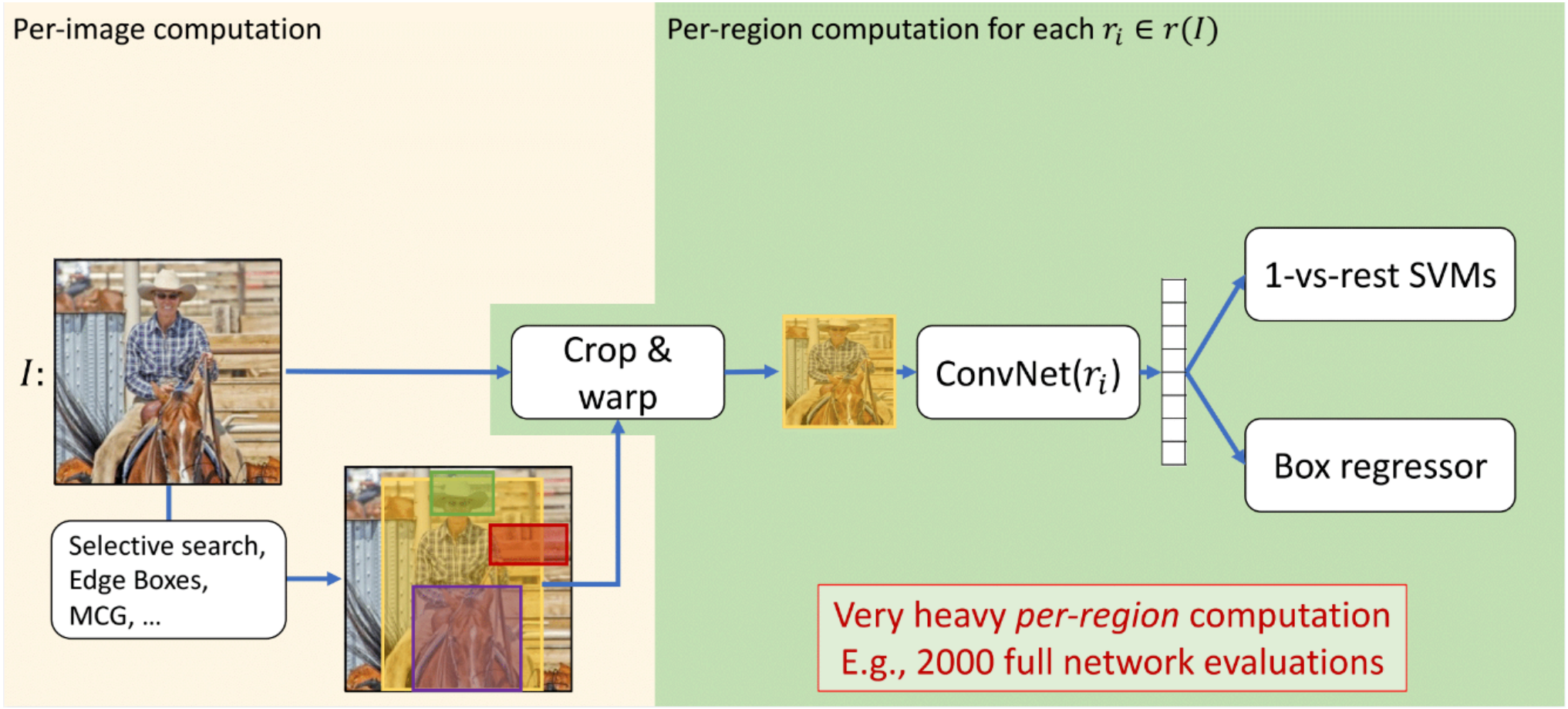
Object Detection Frameworks: R-CNN



Object Detection Frameworks: R-CNN



Object Detection Frameworks: R-CNN



Object Detection Frameworks

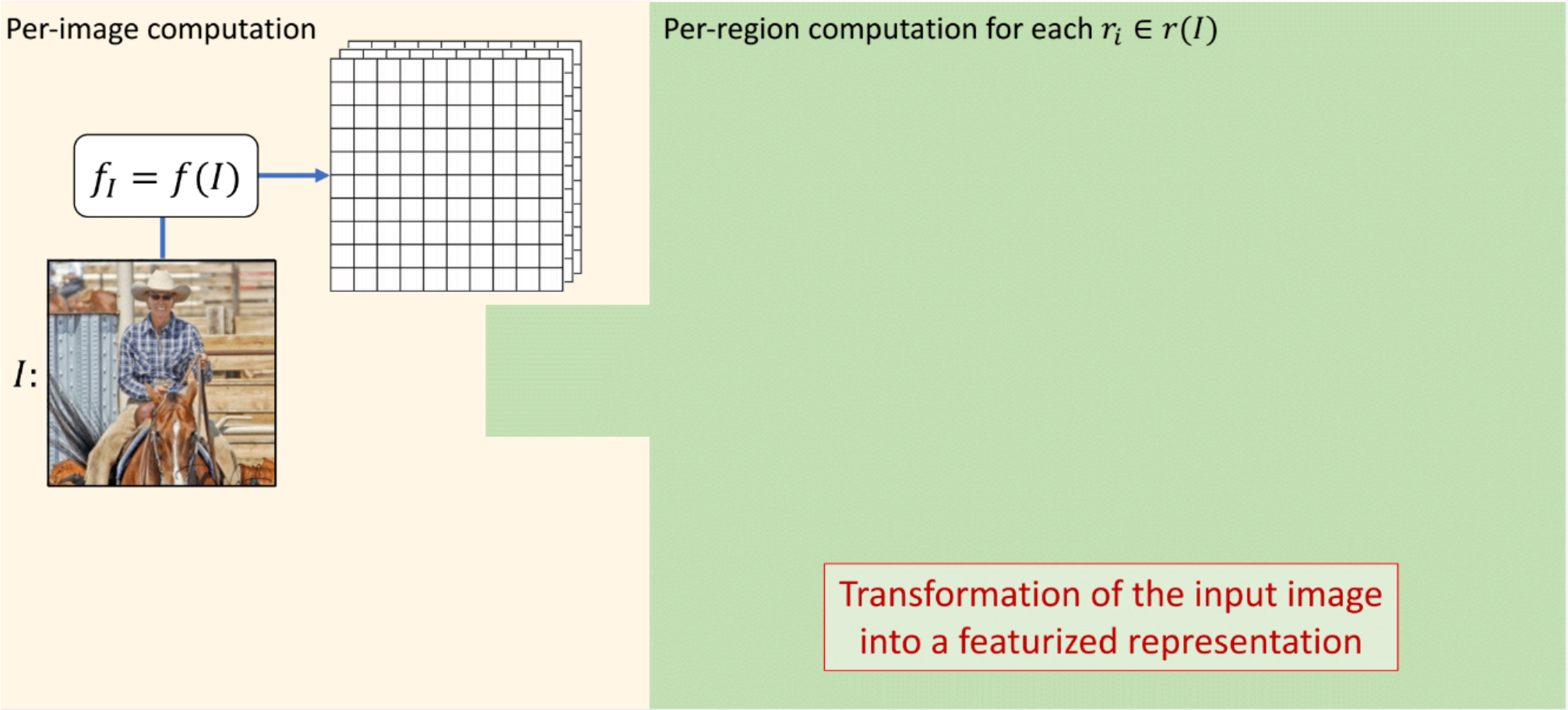
Per-image computation

I:

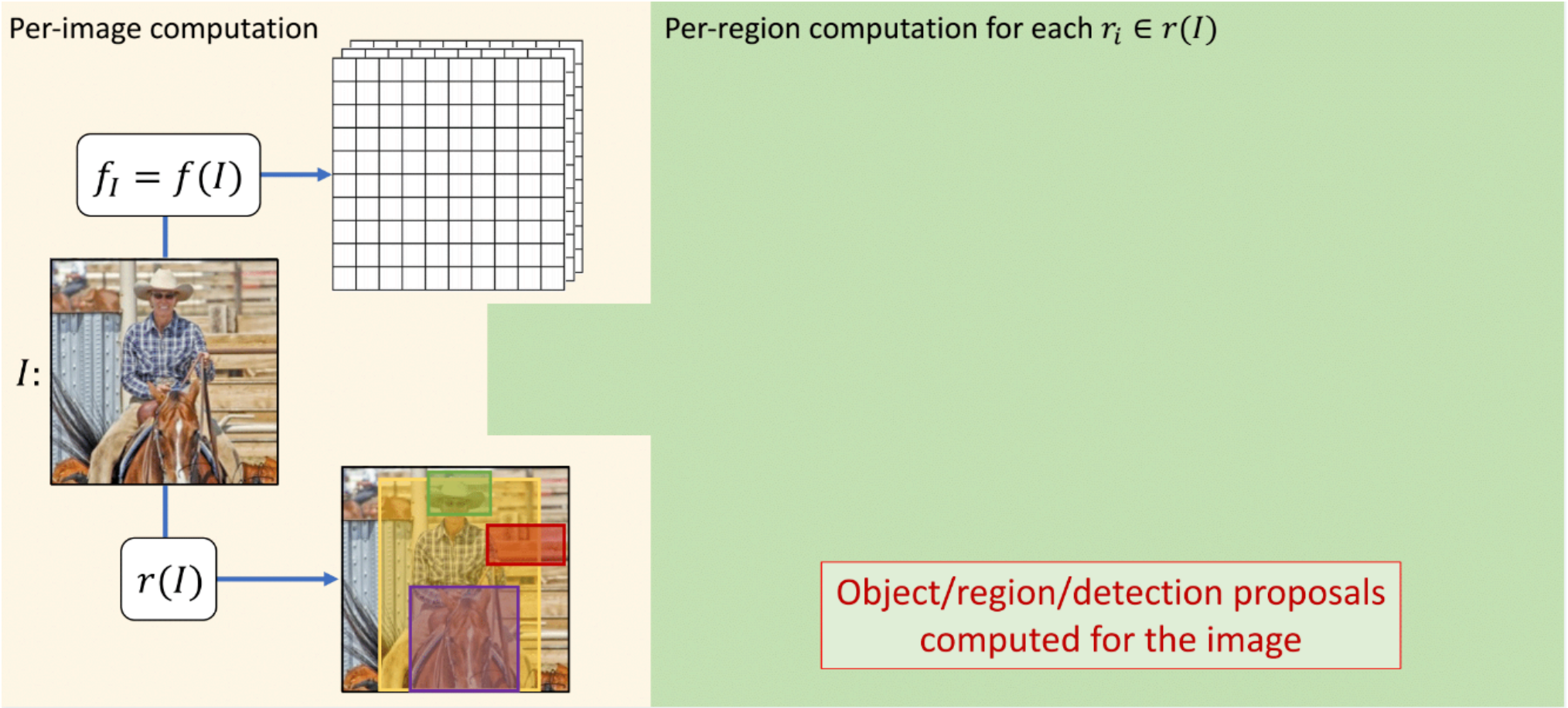
Per-region computation for each $r_i \in r(I)$

Input image
per-image operations | per-region operations

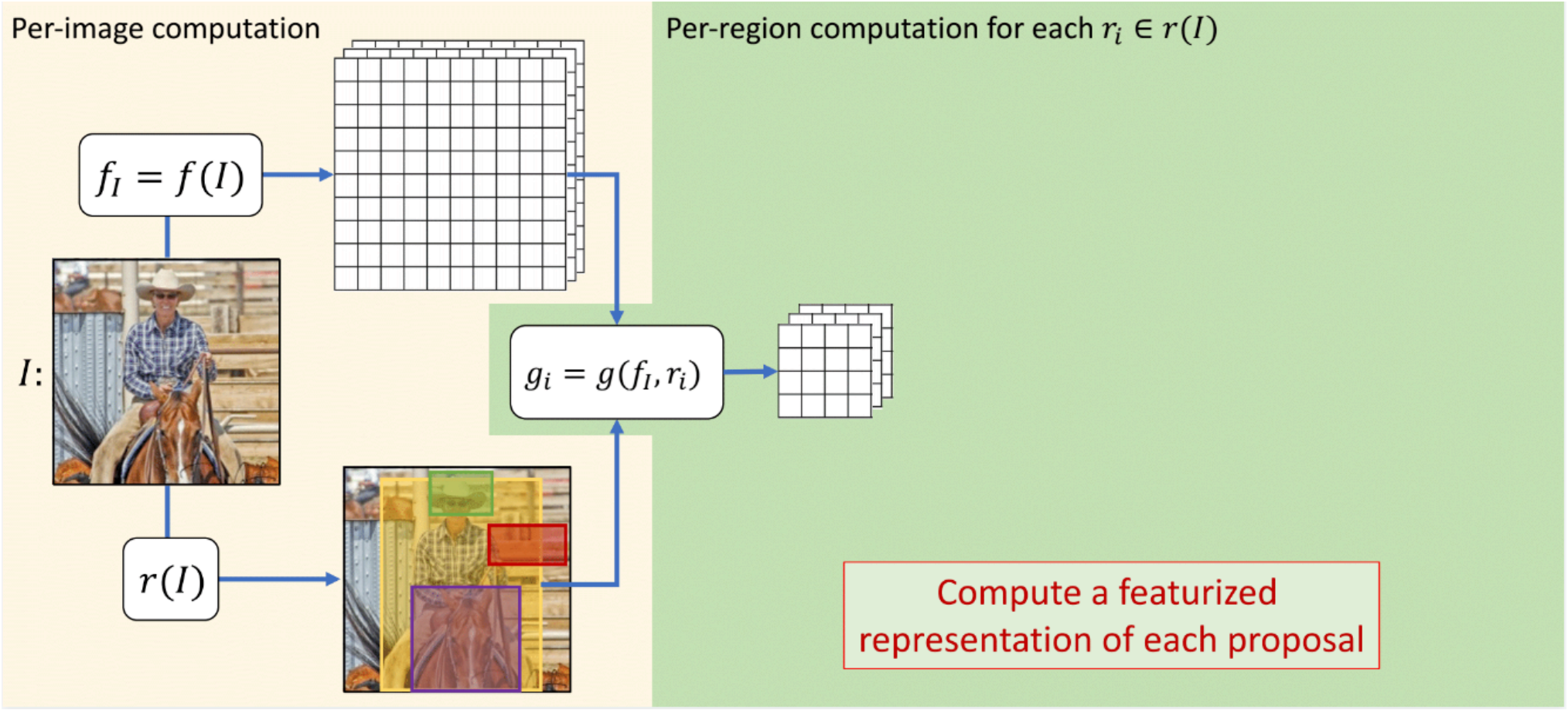
Object Detection Frameworks



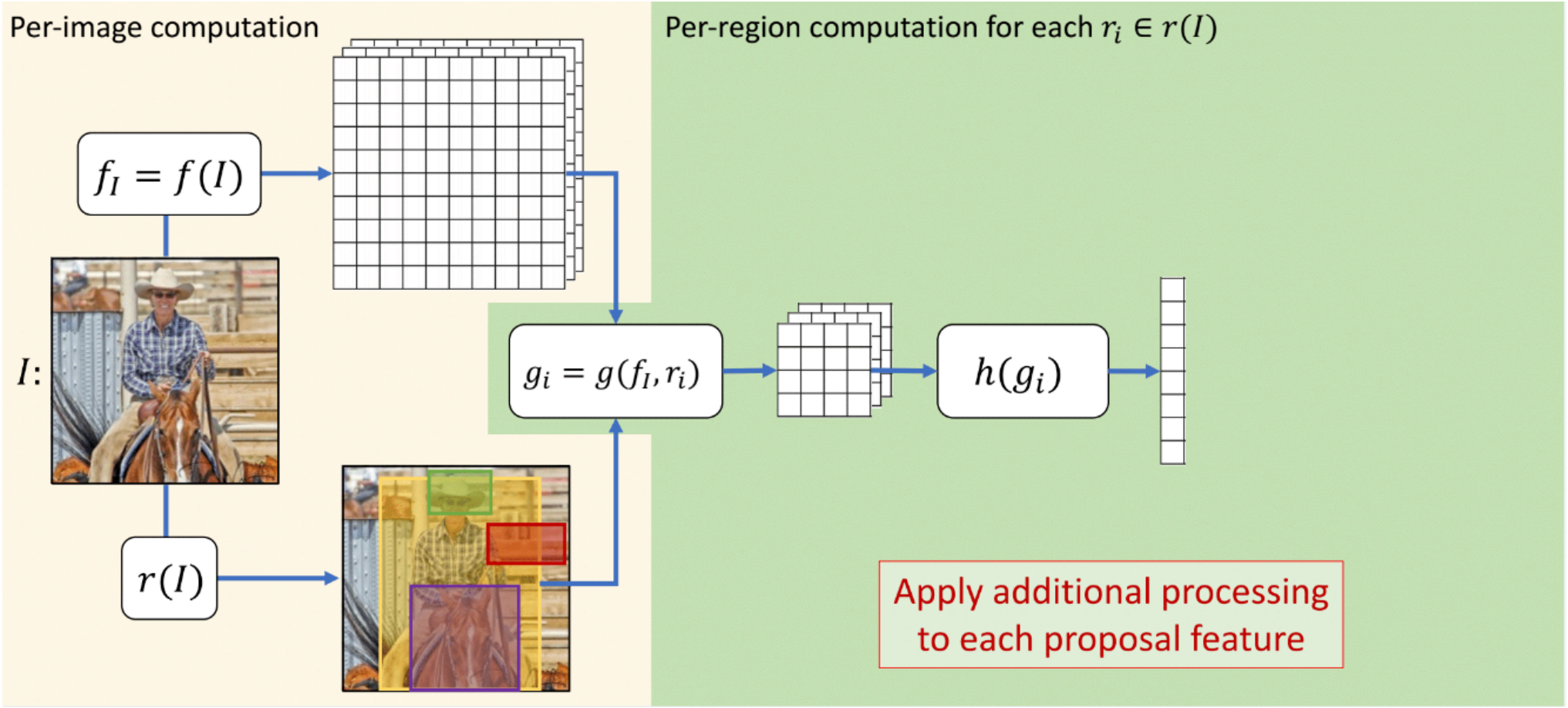
Object Detection Frameworks



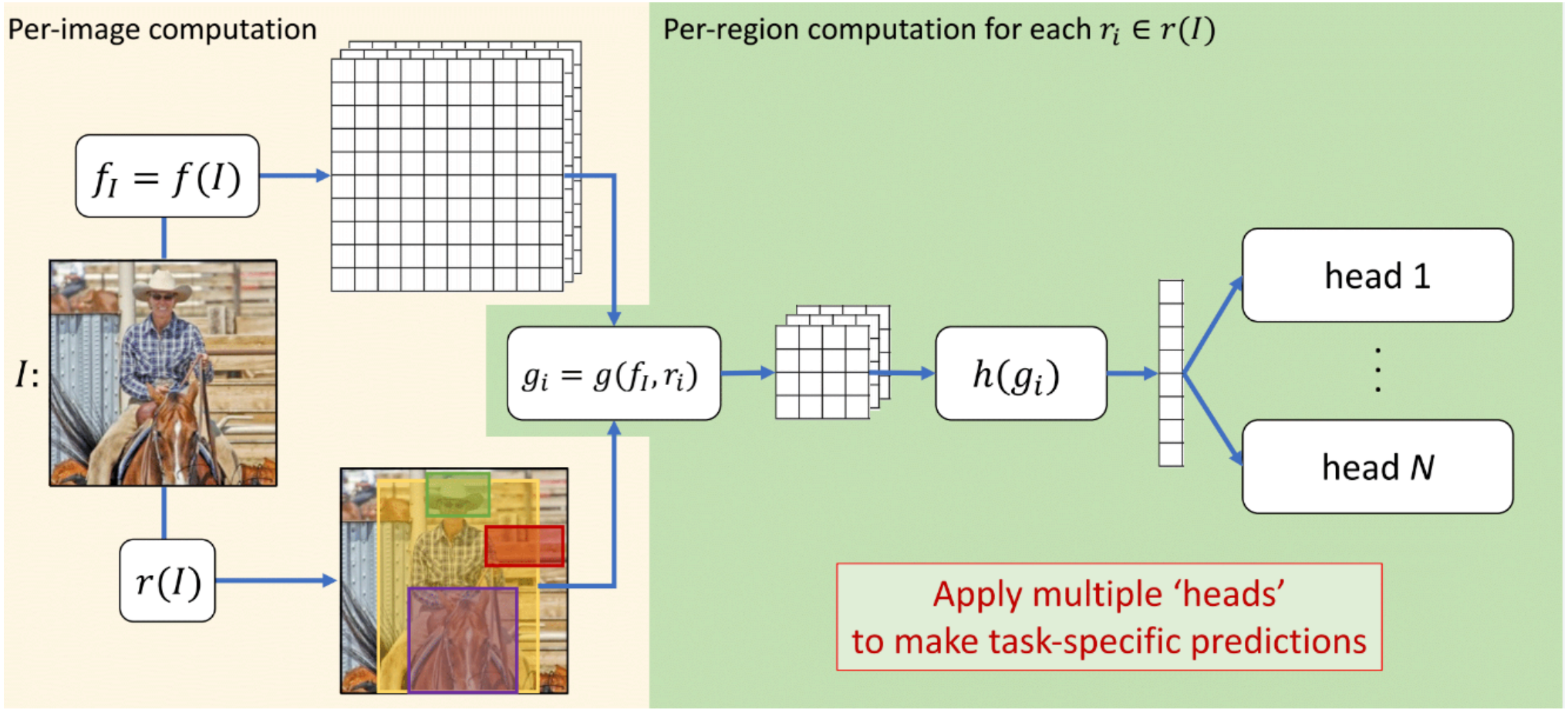
Object Detection Frameworks



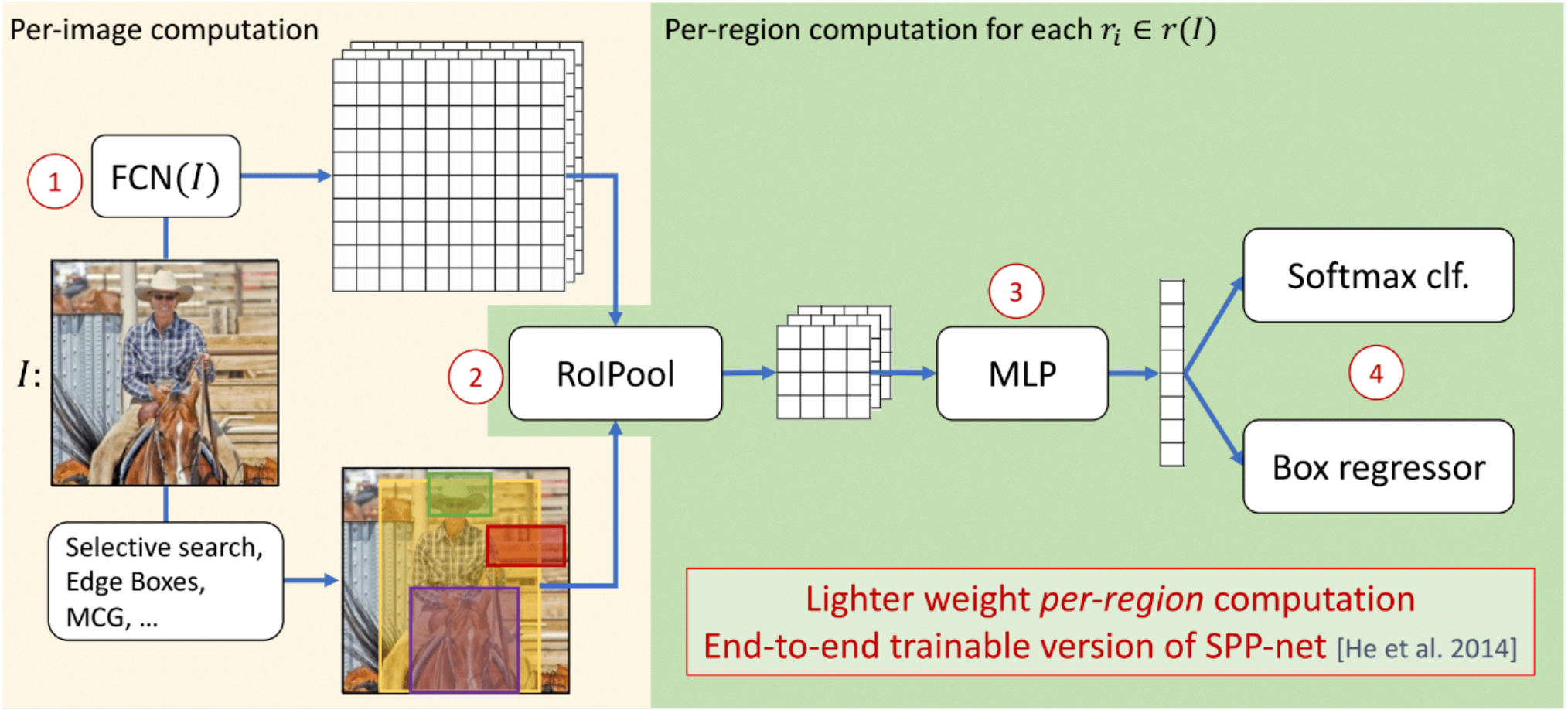
Object Detection Frameworks



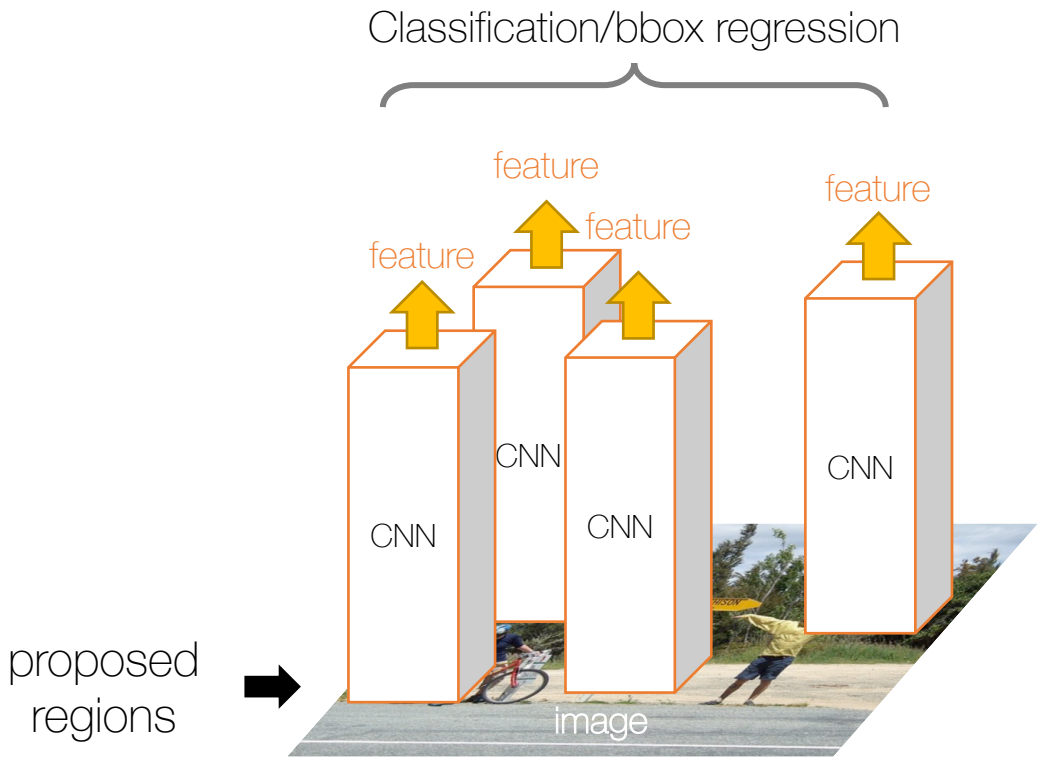
Object Detection Frameworks



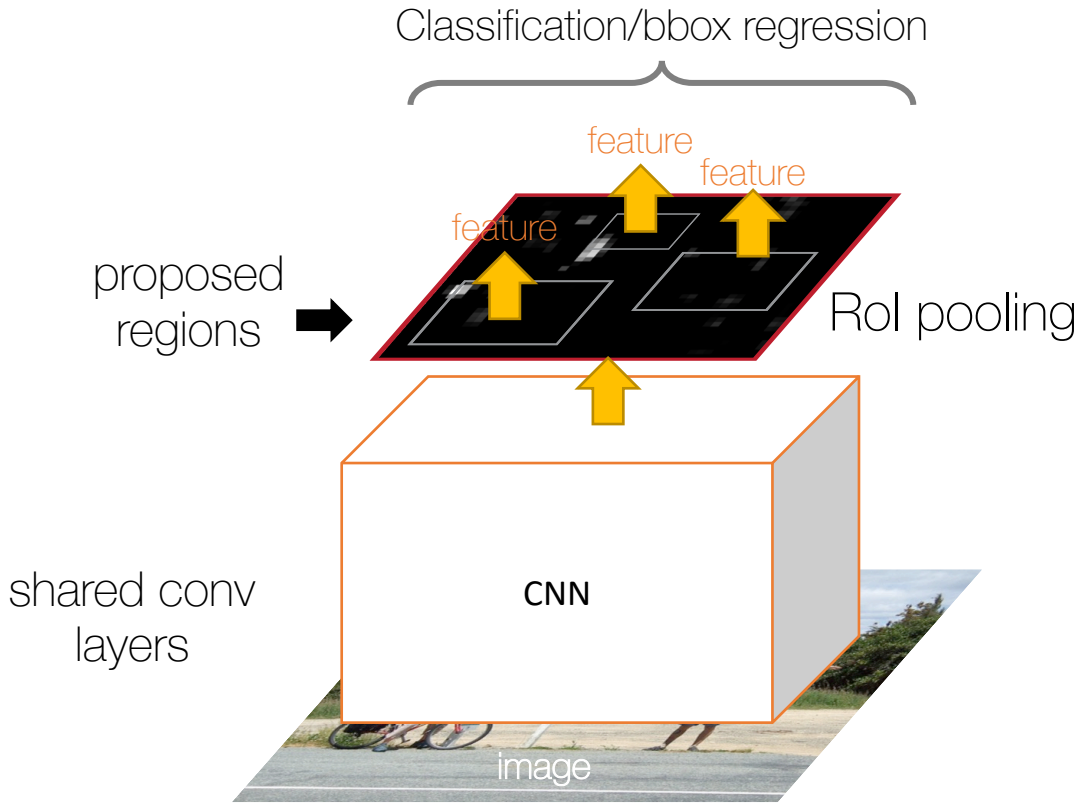
Object Detection Frameworks: Fast R-CNN



Object Detection Frameworks



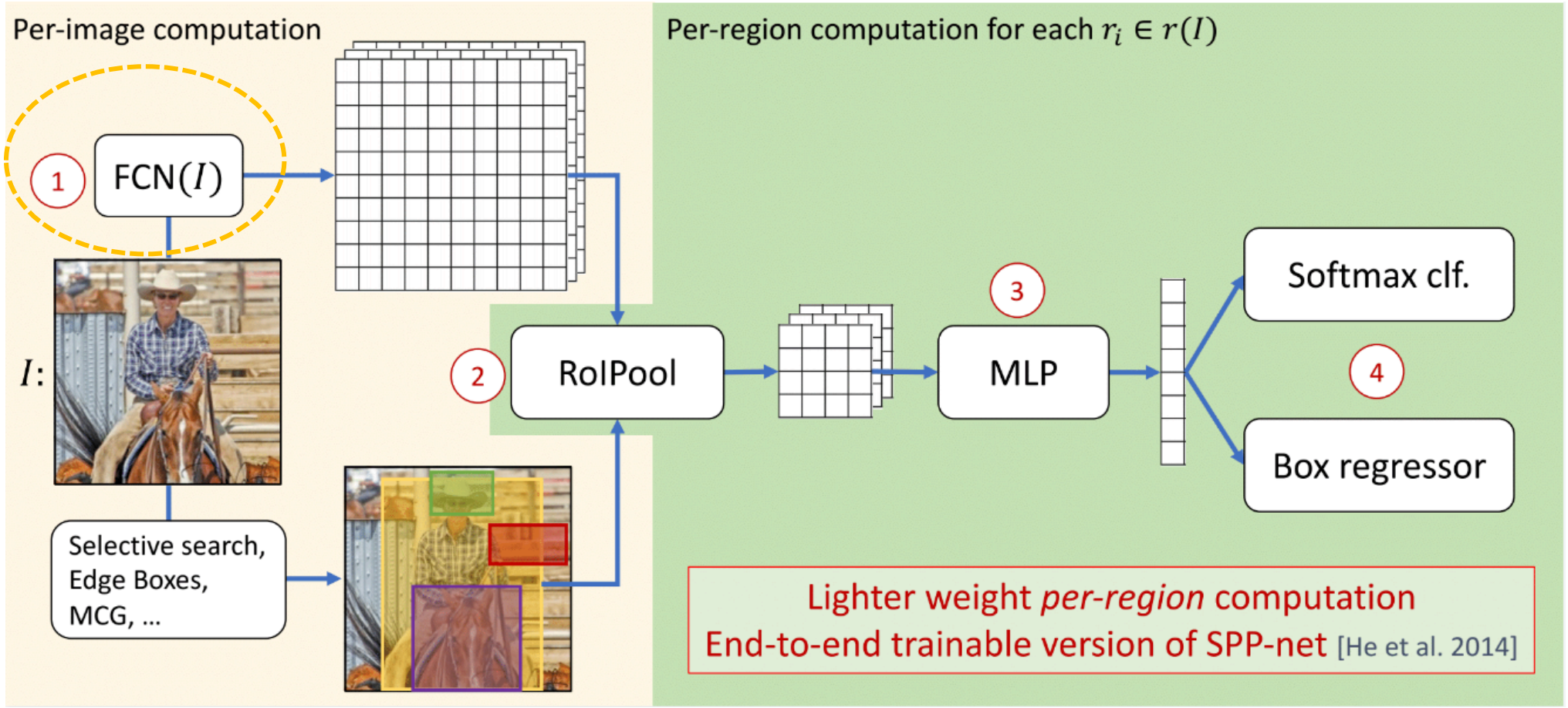
R-CNN



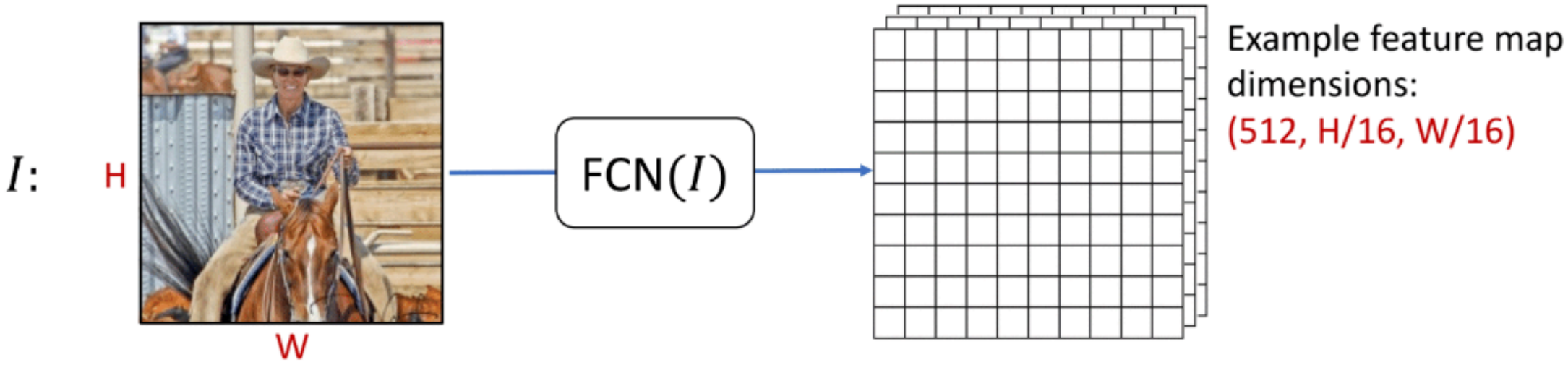
Fast R-CNN

Girshick et al, CVPR 2014 & ICCV 2015

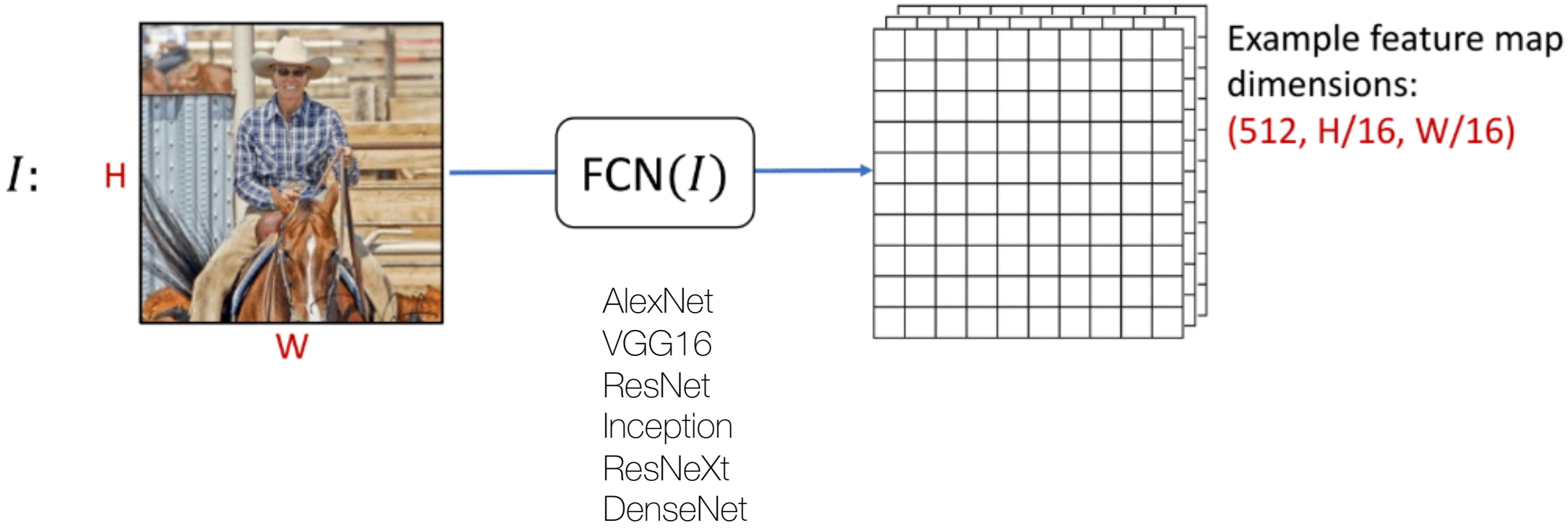
Backbone Networks



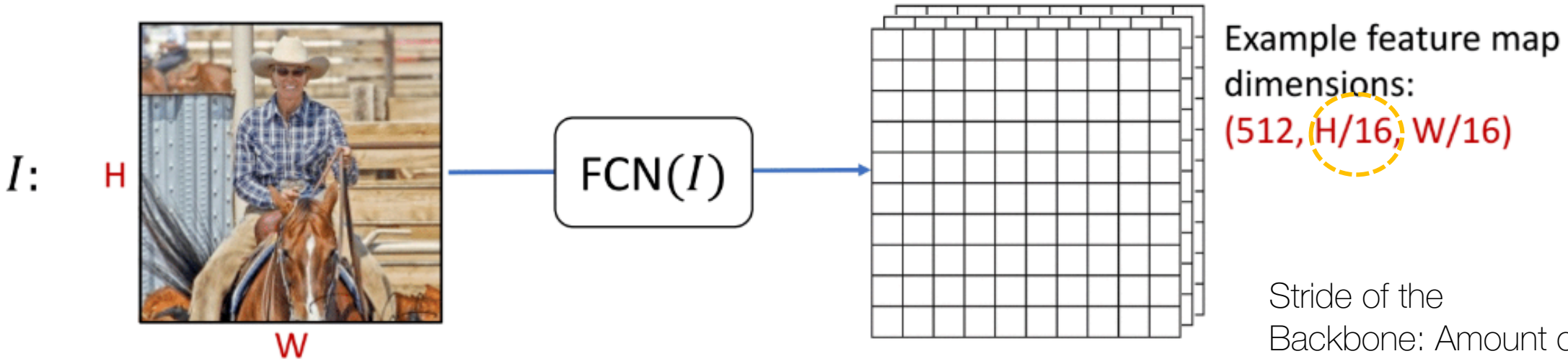
Backbone Networks



Backbone Networks

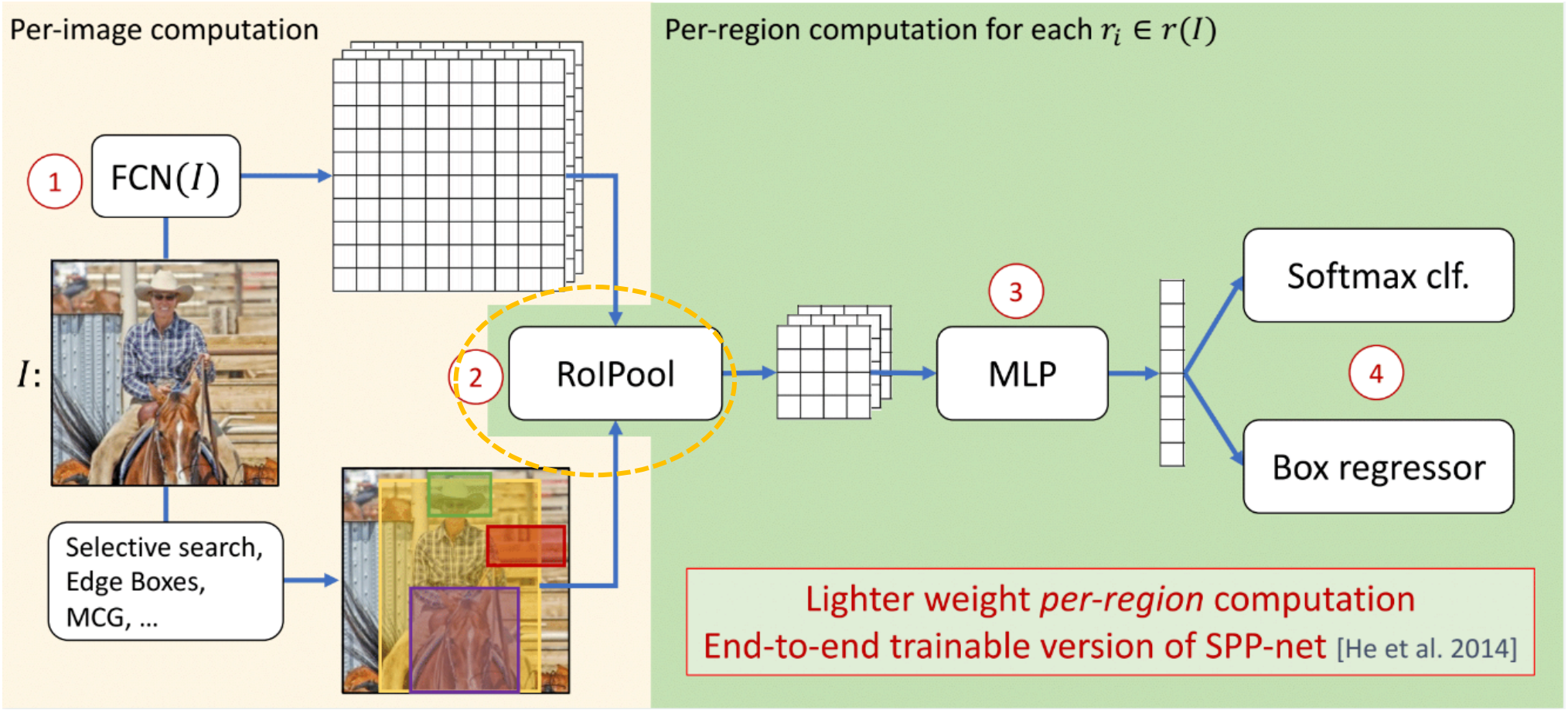


Backbone Networks

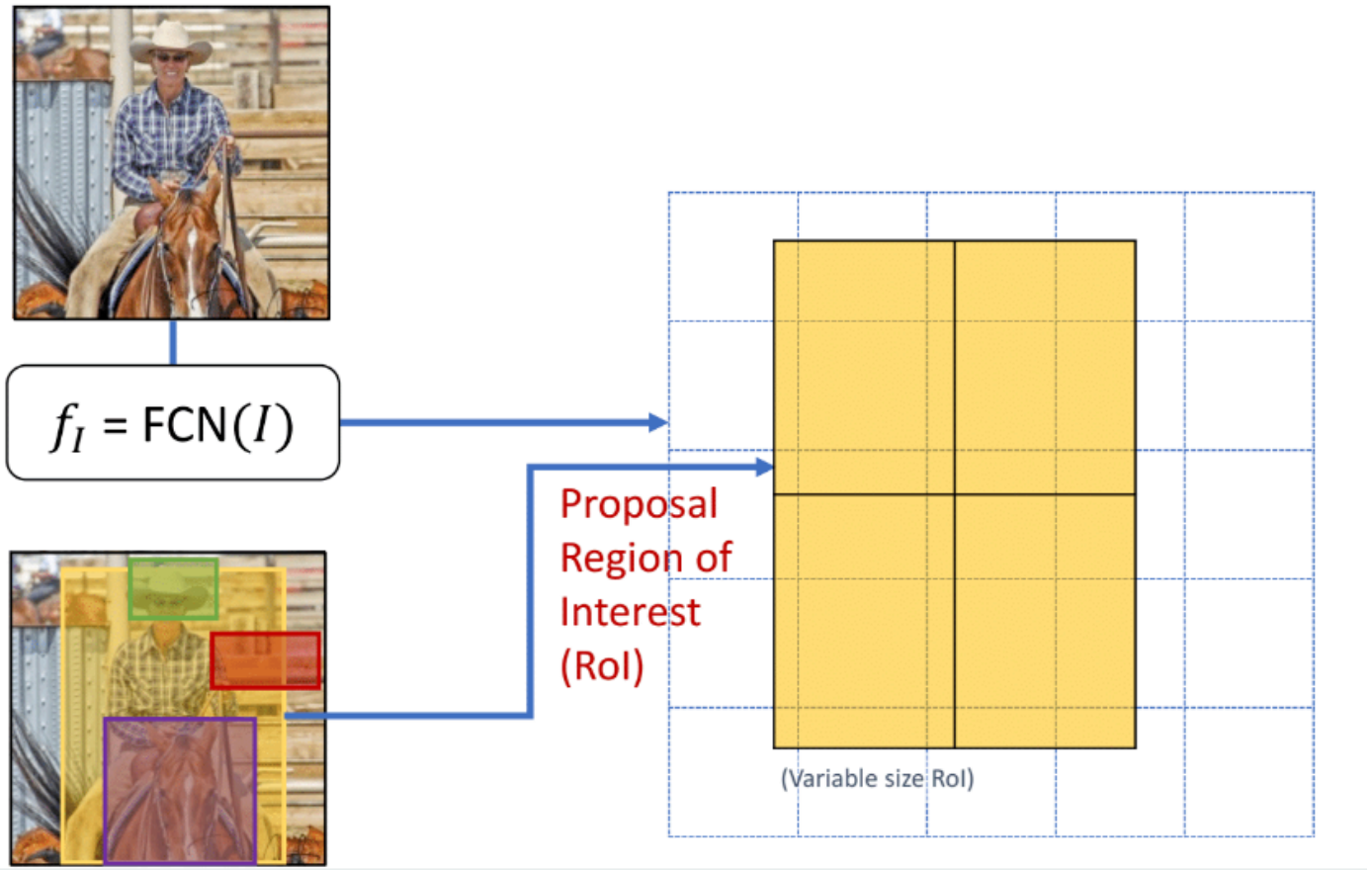


Stride of the Backbone: Amount of strided pooling or convolutions

RoIPool

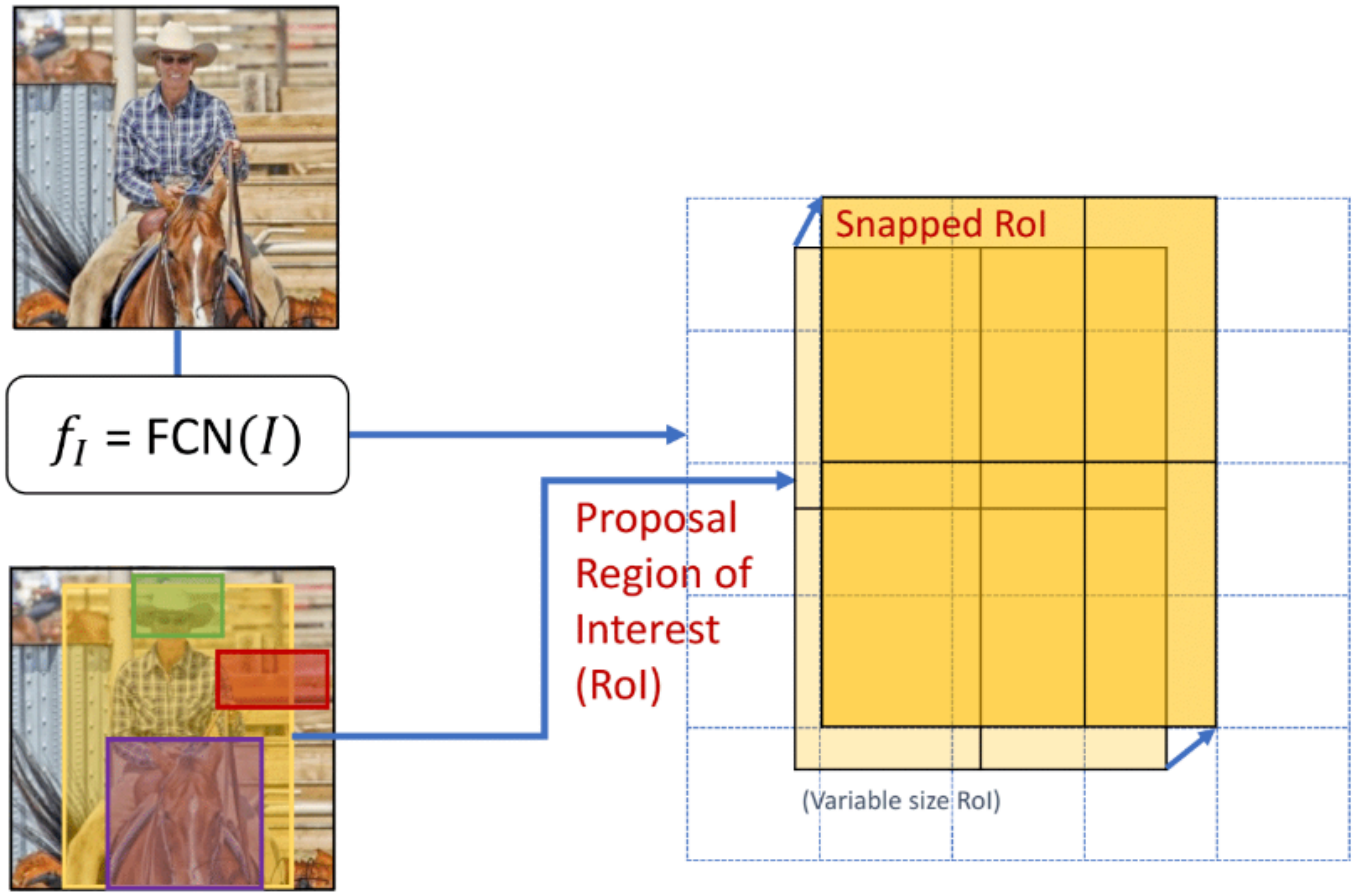

Per-image computation
Per-region computation for each $r_i \in r(I)$
1
FCN(I)
I:
2
RoIPool
3
MLP
Softmax clf.
4
Box regressor
Selective search,
Edge Boxes,
MCG, …
Lighter weight *per-region* computation
End-to-end trainable version of SPP-net [He et al. 2014]

RoIPool on Each Proposal



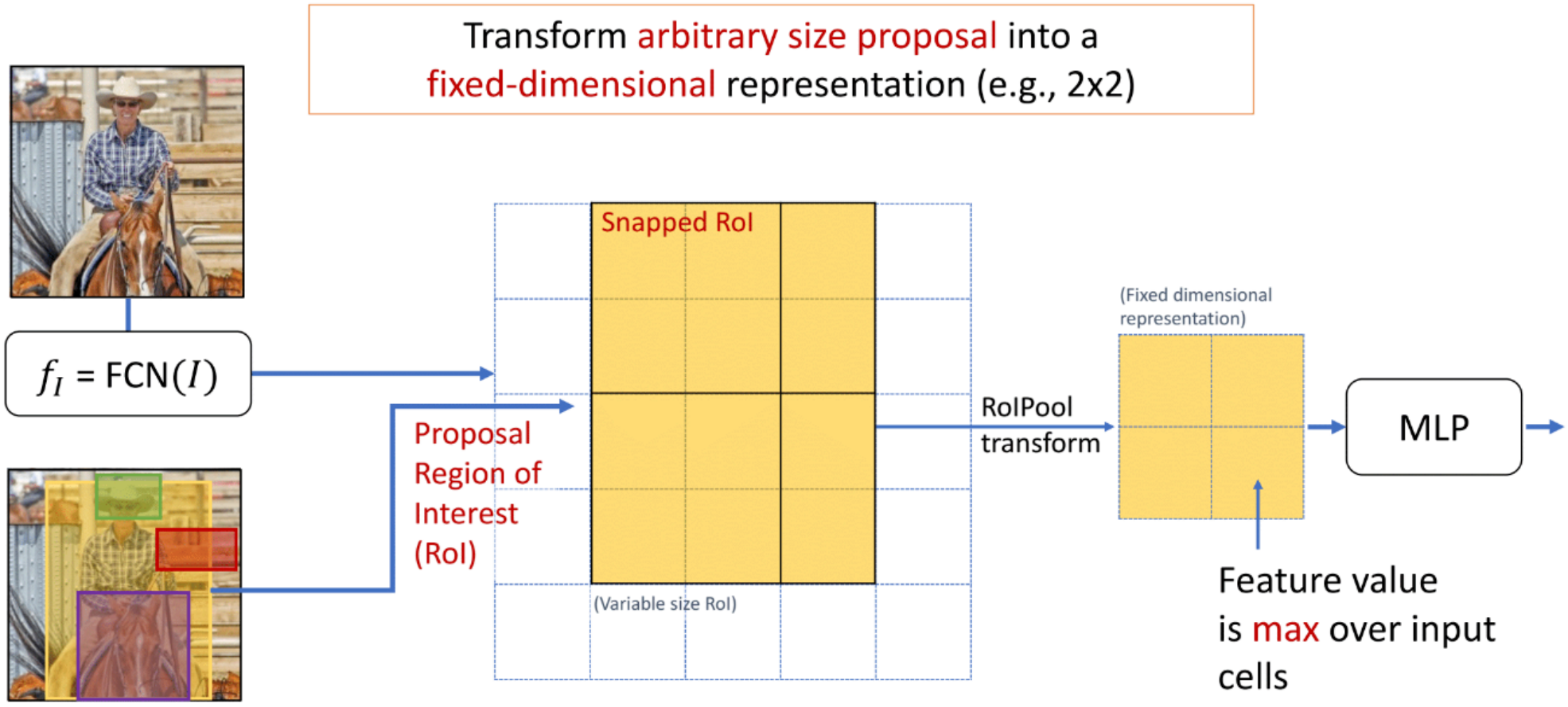
Innovation in SPP-Net [He et al. 2014]

RoIPool on Each Proposal



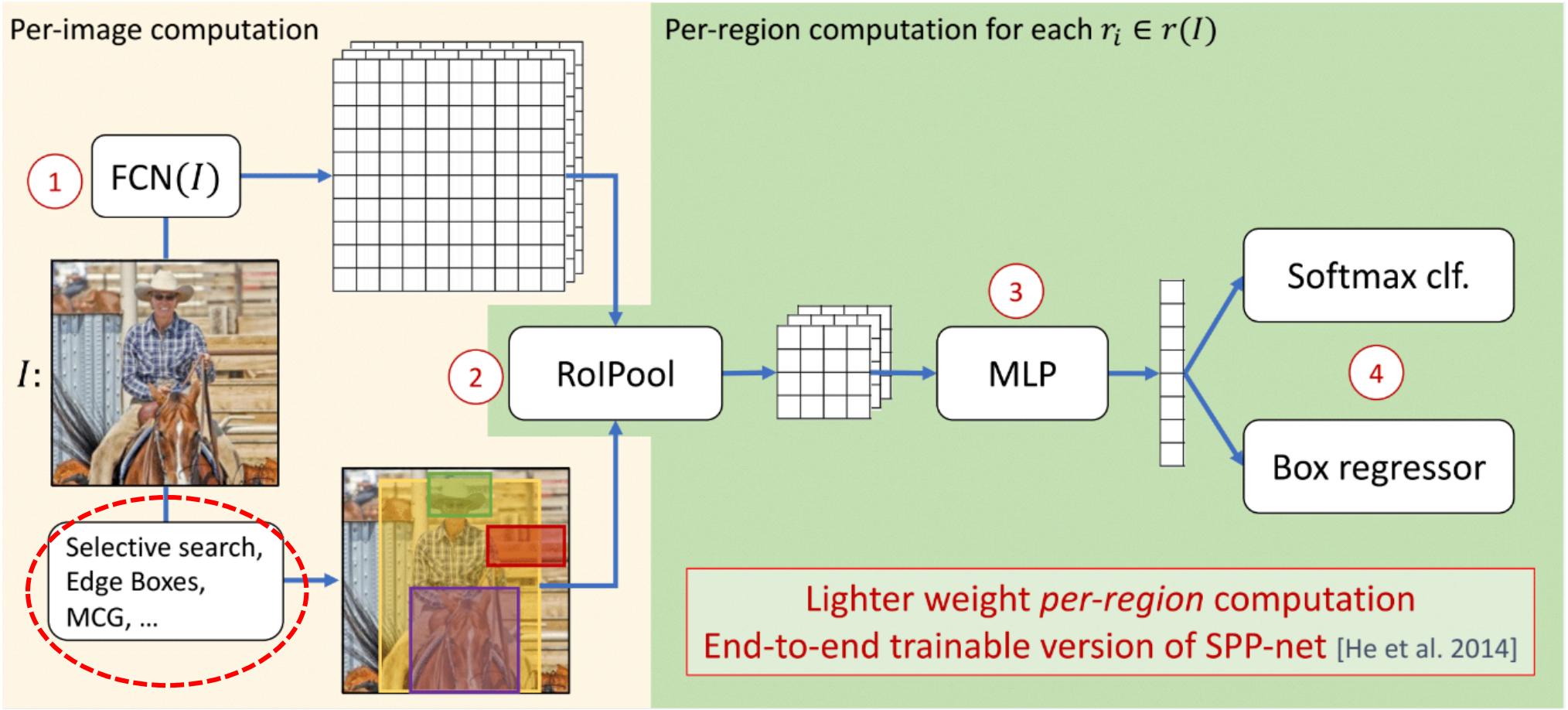
Innovation in SPP-Net [He et al. 2014]

RoIPool on Each Proposal

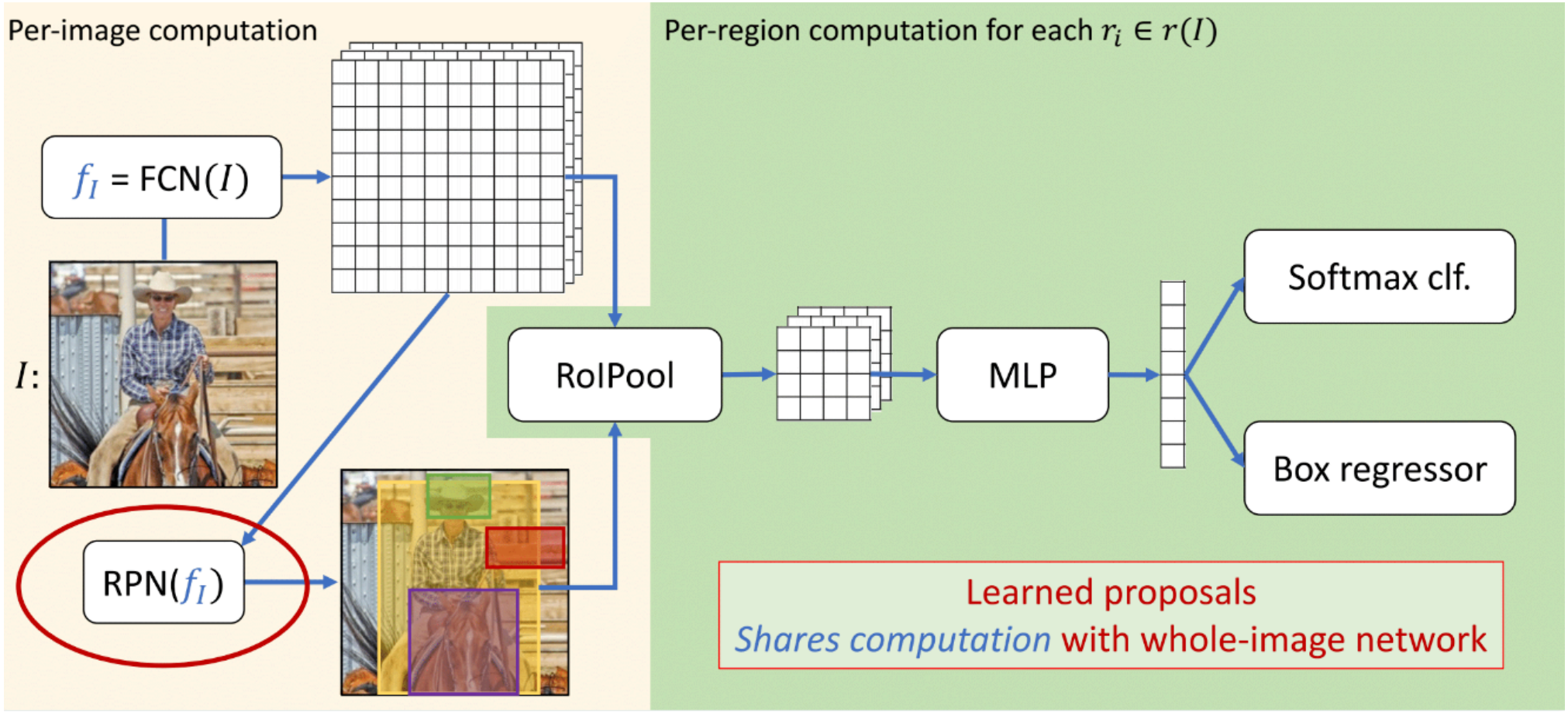


Innovation in SPP-Net [He et al. 2014]

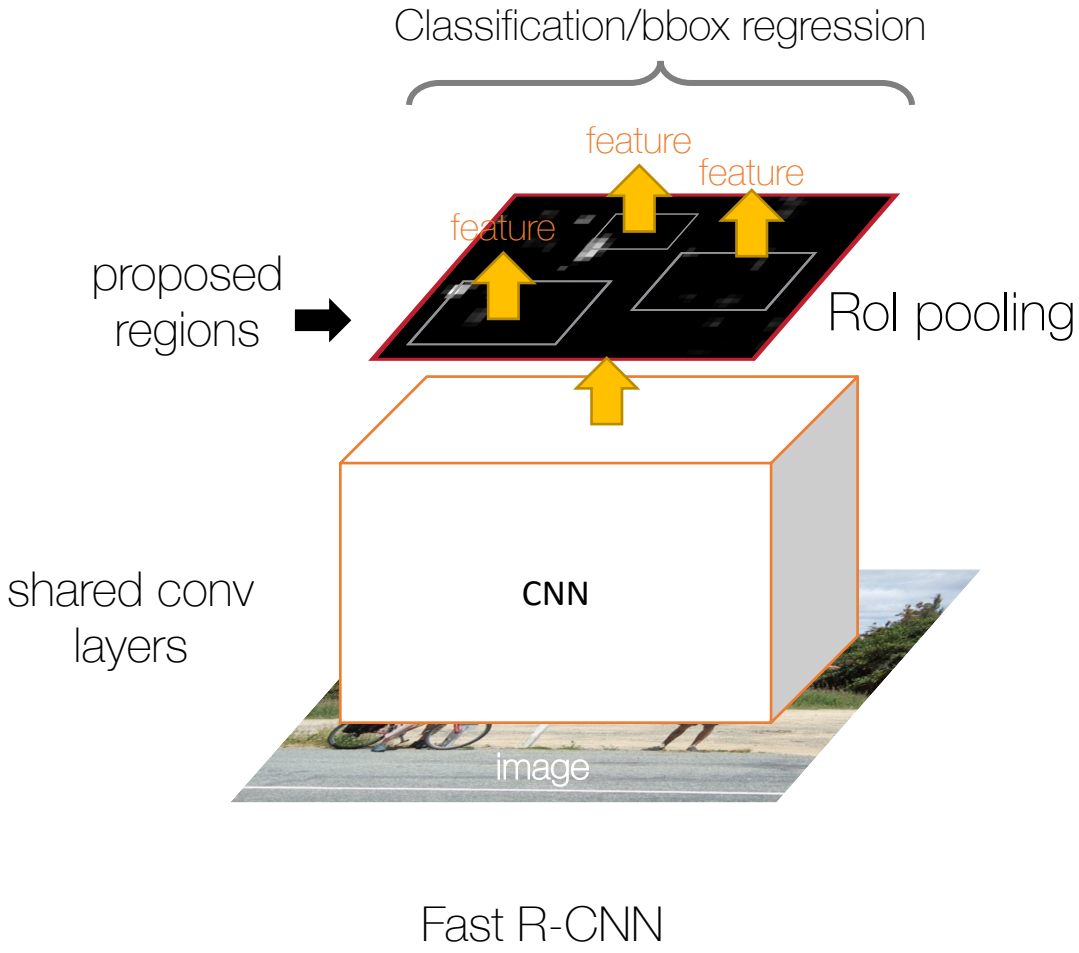
The Problem with Fast R-CNN



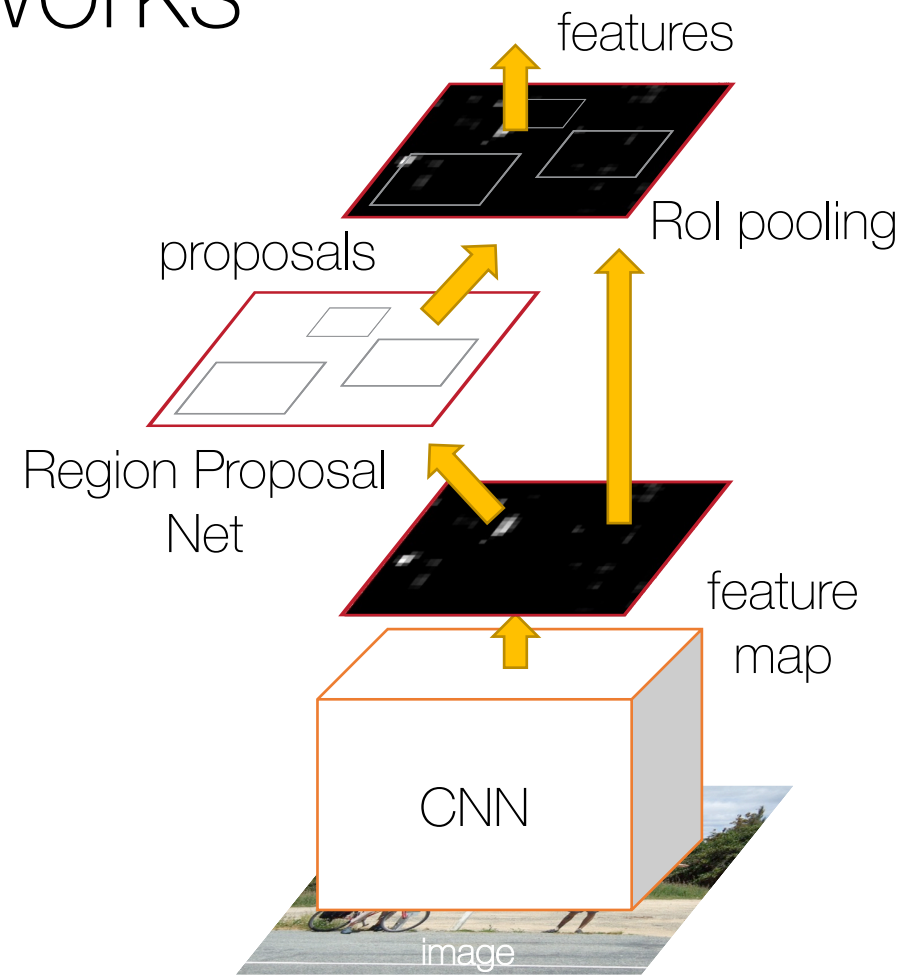
Object Detection Frameworks: Faster R-CNN



Object Detection Frameworks



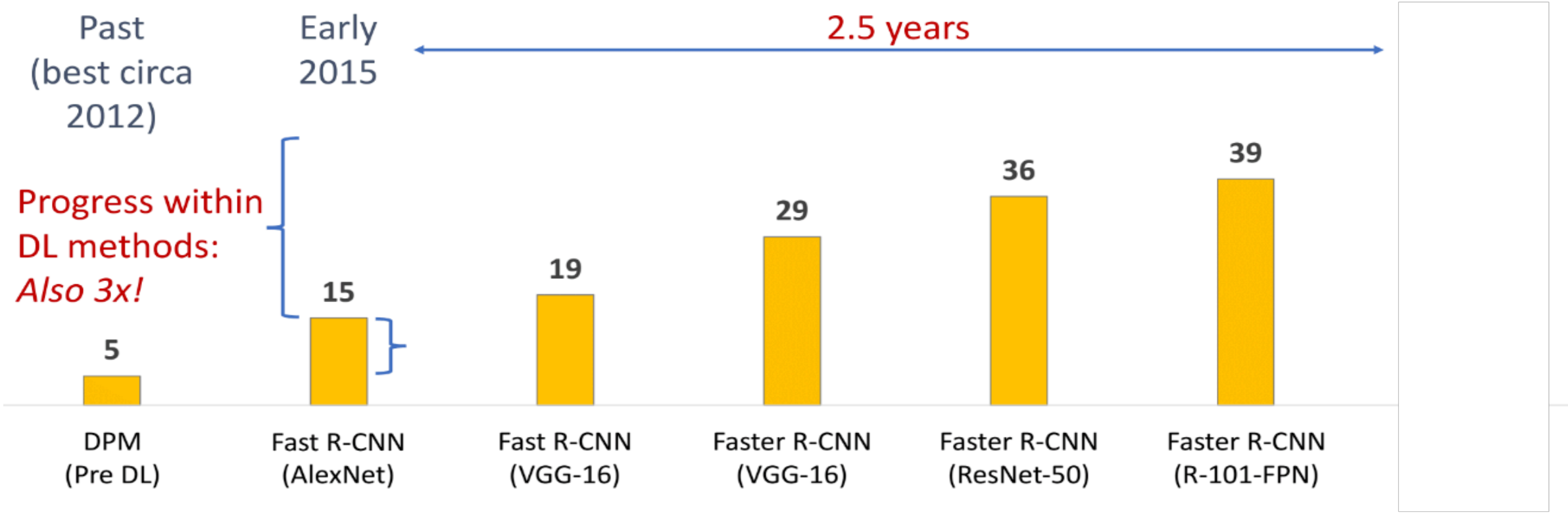
Fast R-CNN



Faster R-CNN

Girshick et al., CVPR 2014 & ICCV 2015, Ren et al., NIPS 2015

Performance through Better Detector Design and Better Backbones



Instance Segmentation

Task

- Assume C object classes
- Segment all object instances in an image

True Positives

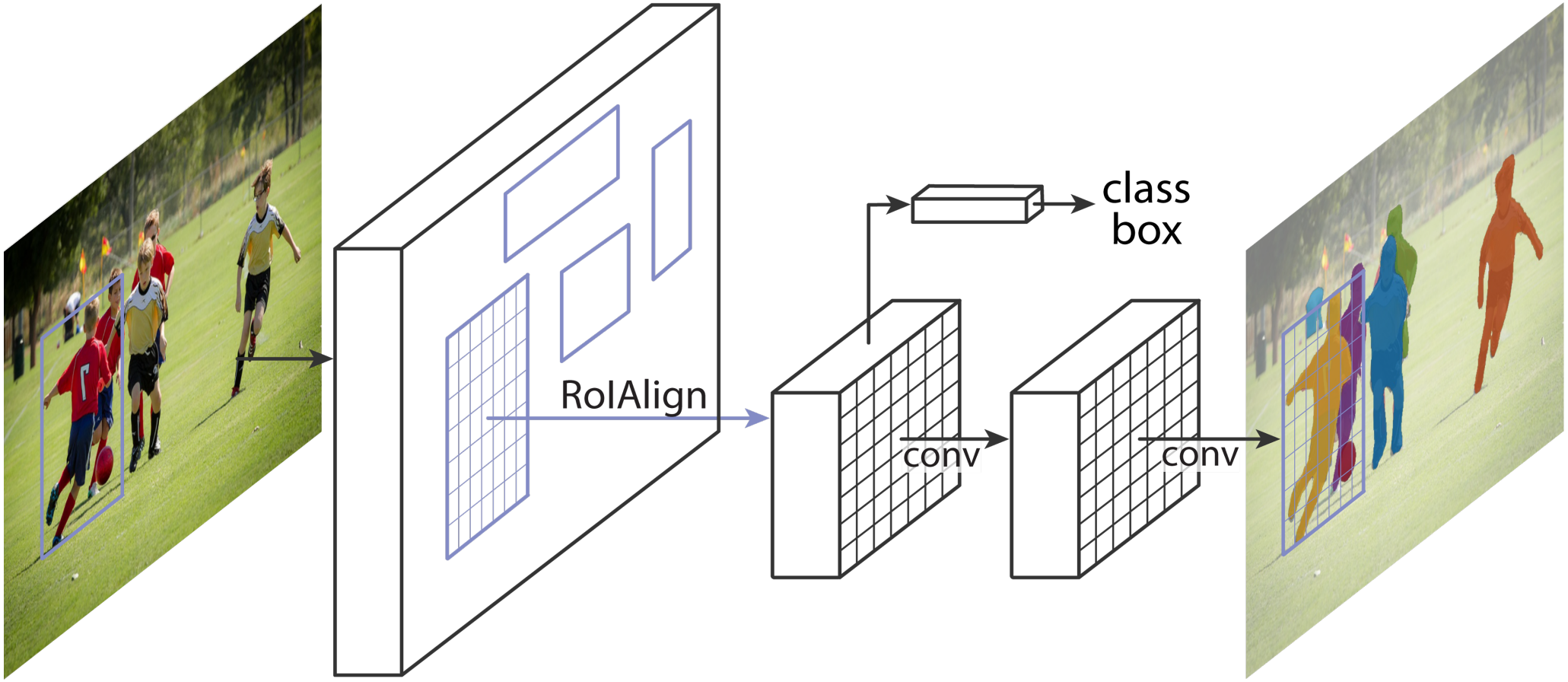
- A detection m is a true positive if maskov(m, gt) > 0.5 and class(m) = class(gt)

False Positives

- Mislocalized and misclassified detections
- Duplicate detections are penalized

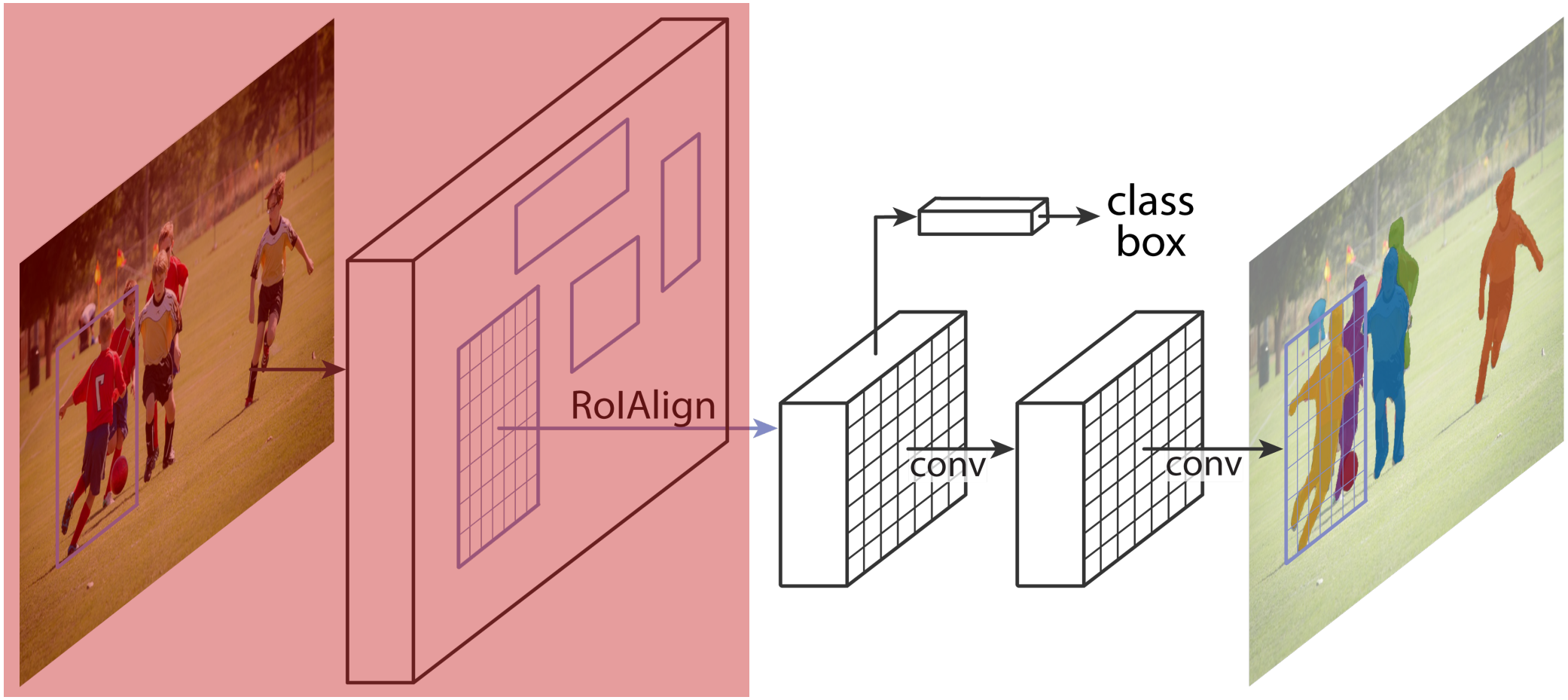
Mask R-CNN

Mask R-CNN



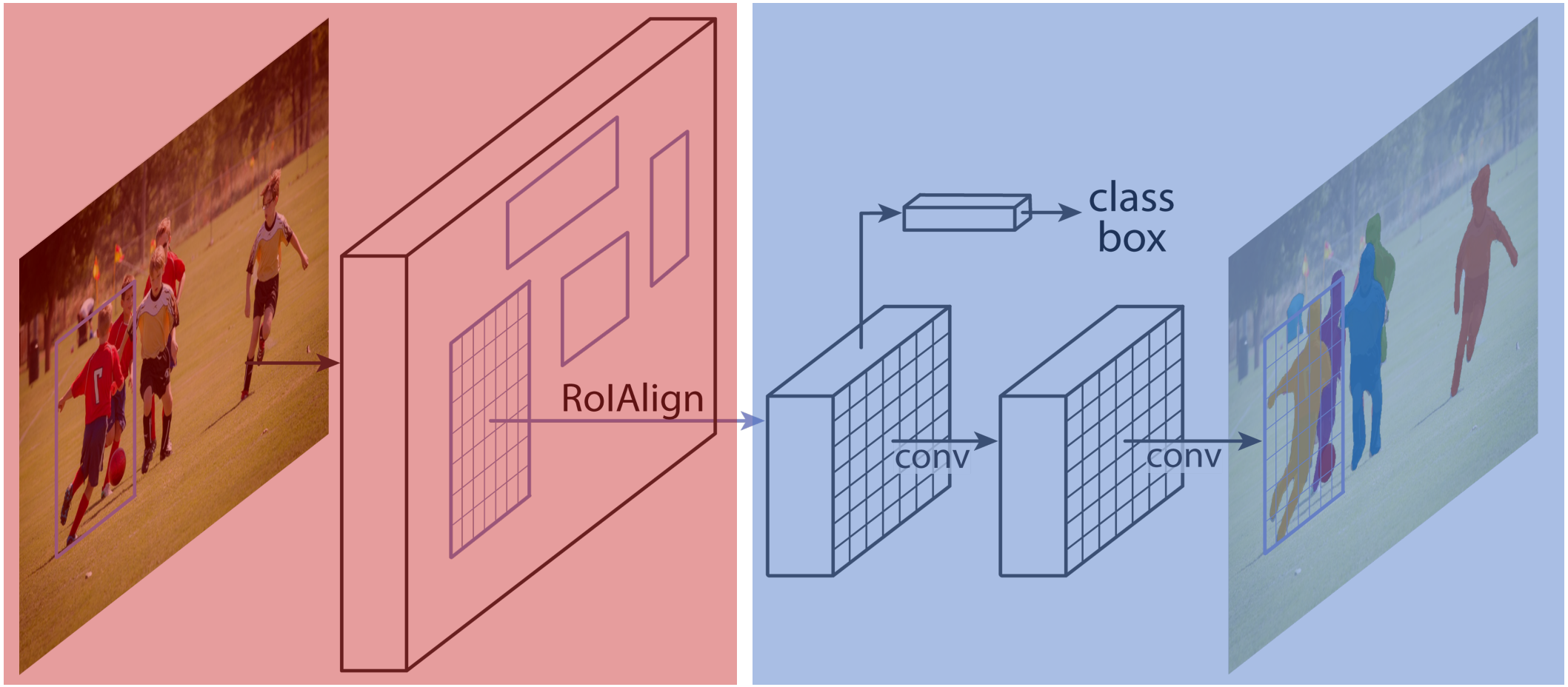
He, Gkioxari, Dollar, Girshick, ICCV 2017

Mask R-CNN



CNN applied to
entire image

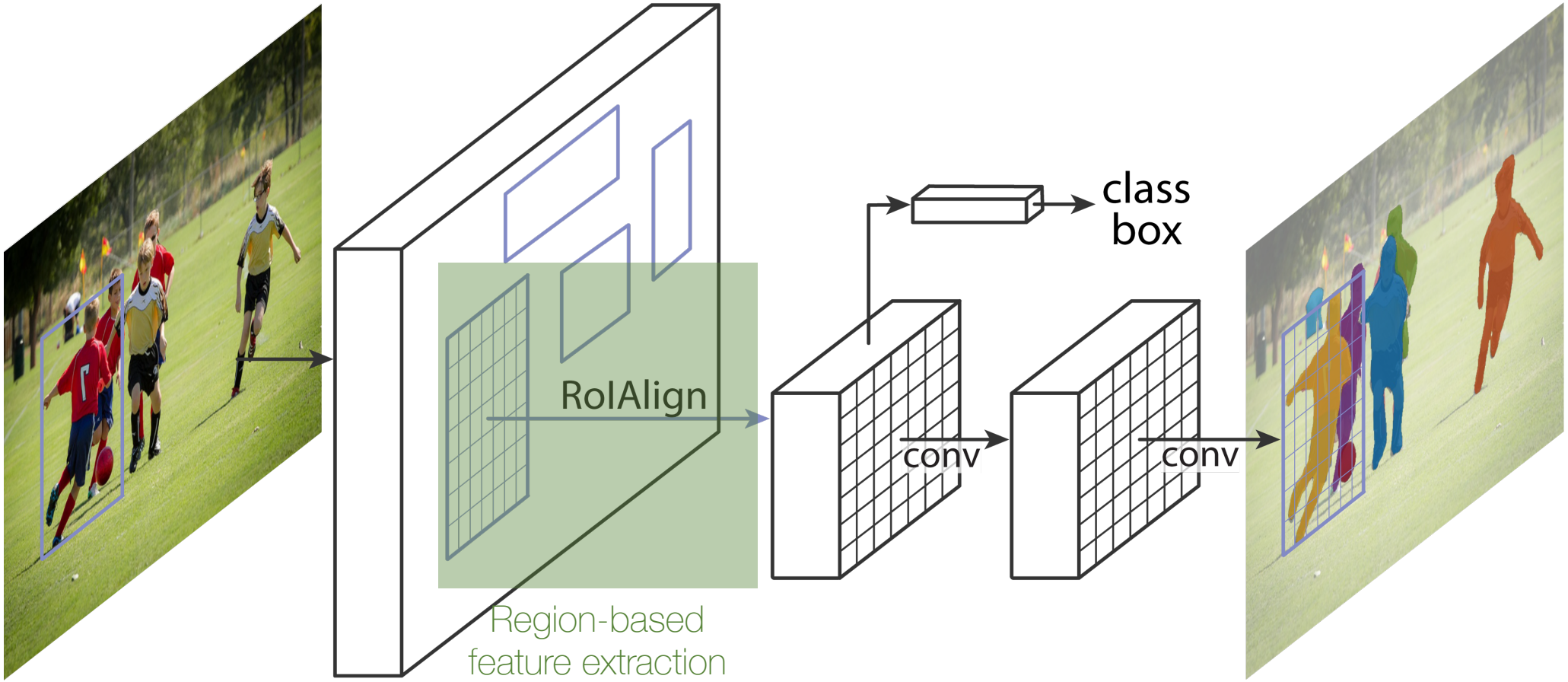
Mask R-CNN



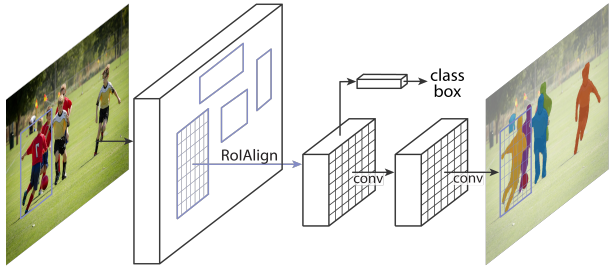
CNN applied to entire image

Region-wise segmentation subnetwork

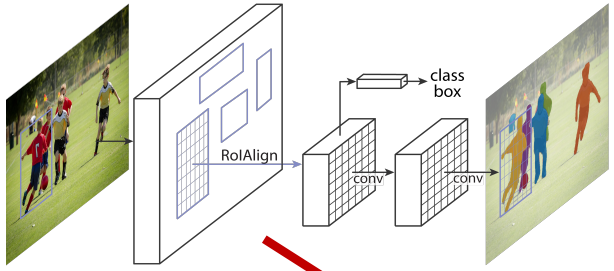
RoIAlign

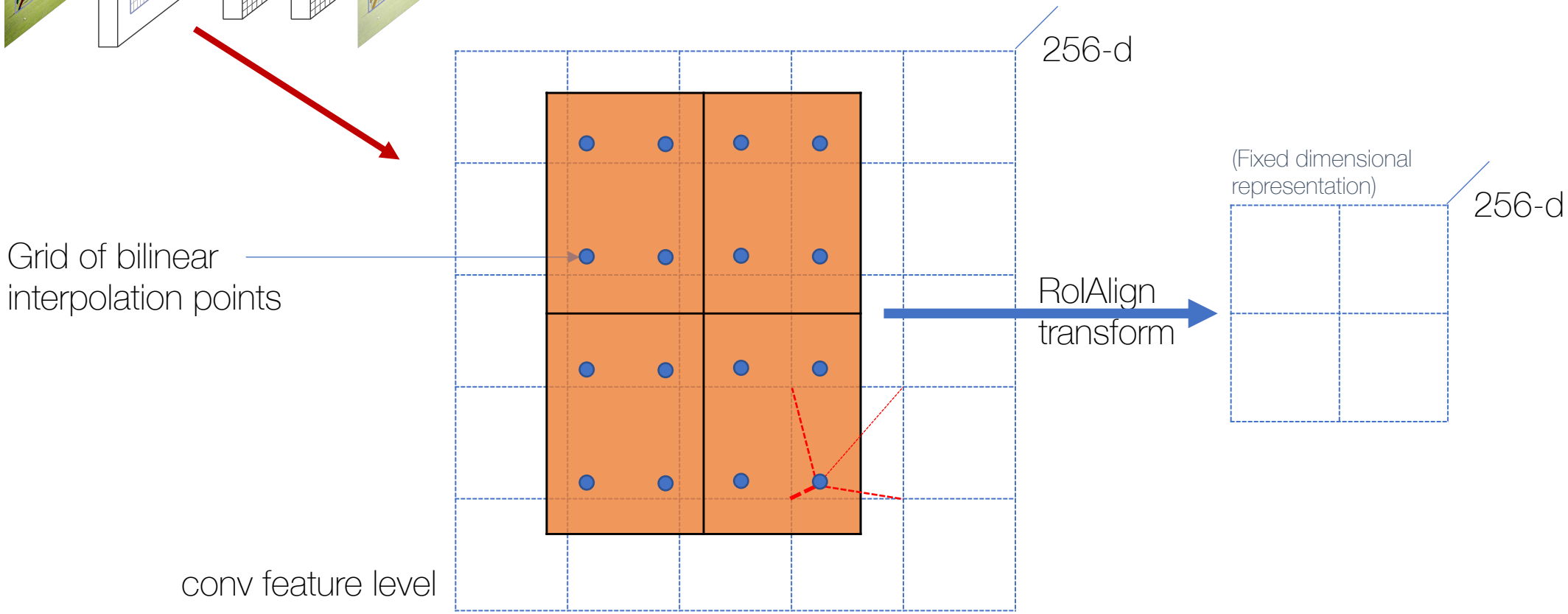

RoIAlign
class
box
conv
conv
Region-based
feature extraction

RoIAlign



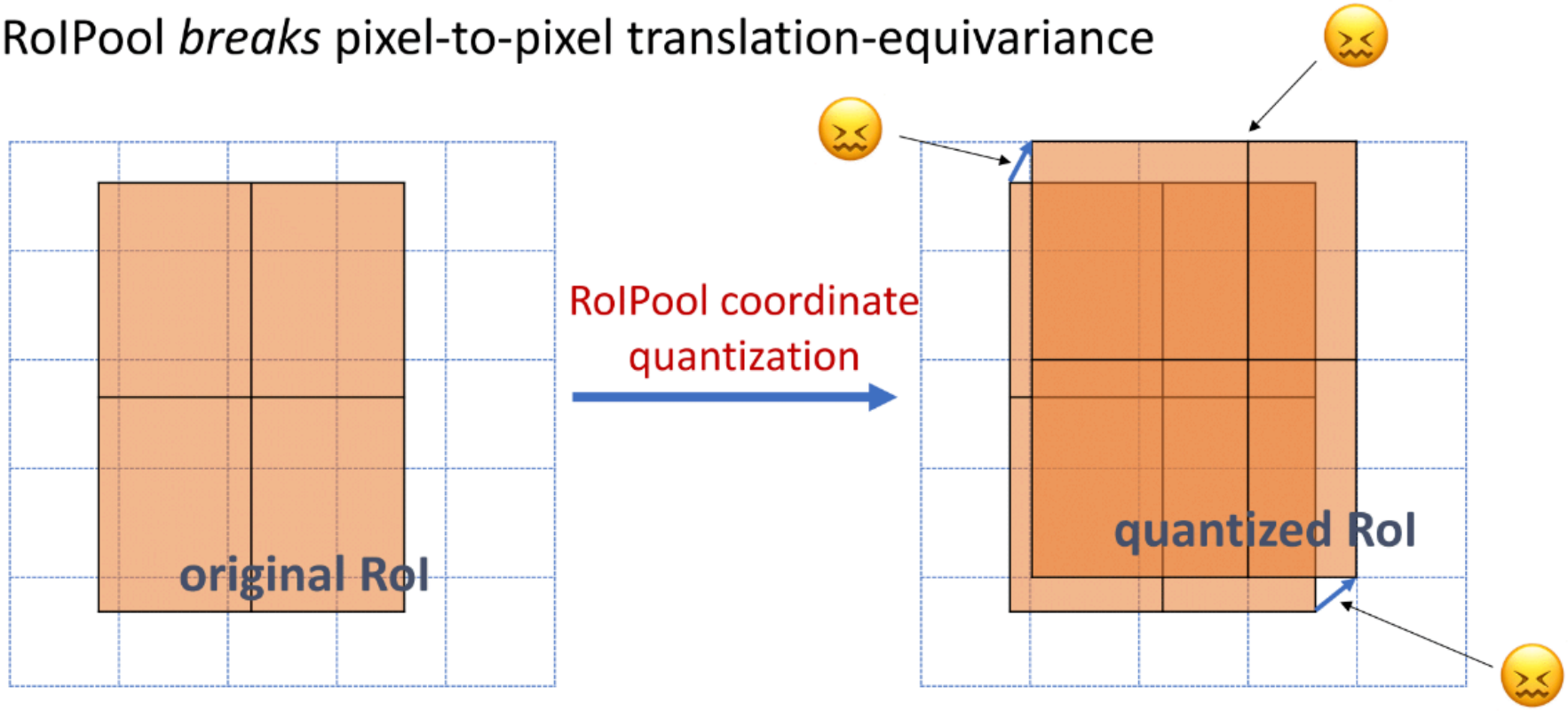
RoIAlign


RoIAlign
class
box
conv
conv

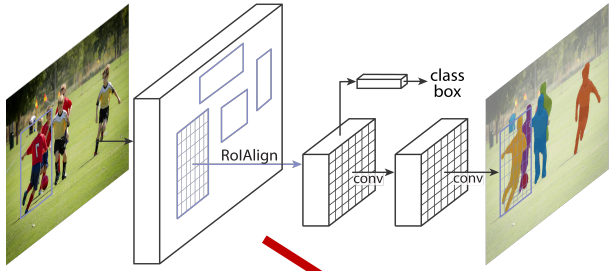

256-d
Grid of bilinear
interpolation points
RoIAlign
transform
(Fixed dimensional
representation)
256-d
conv feature level

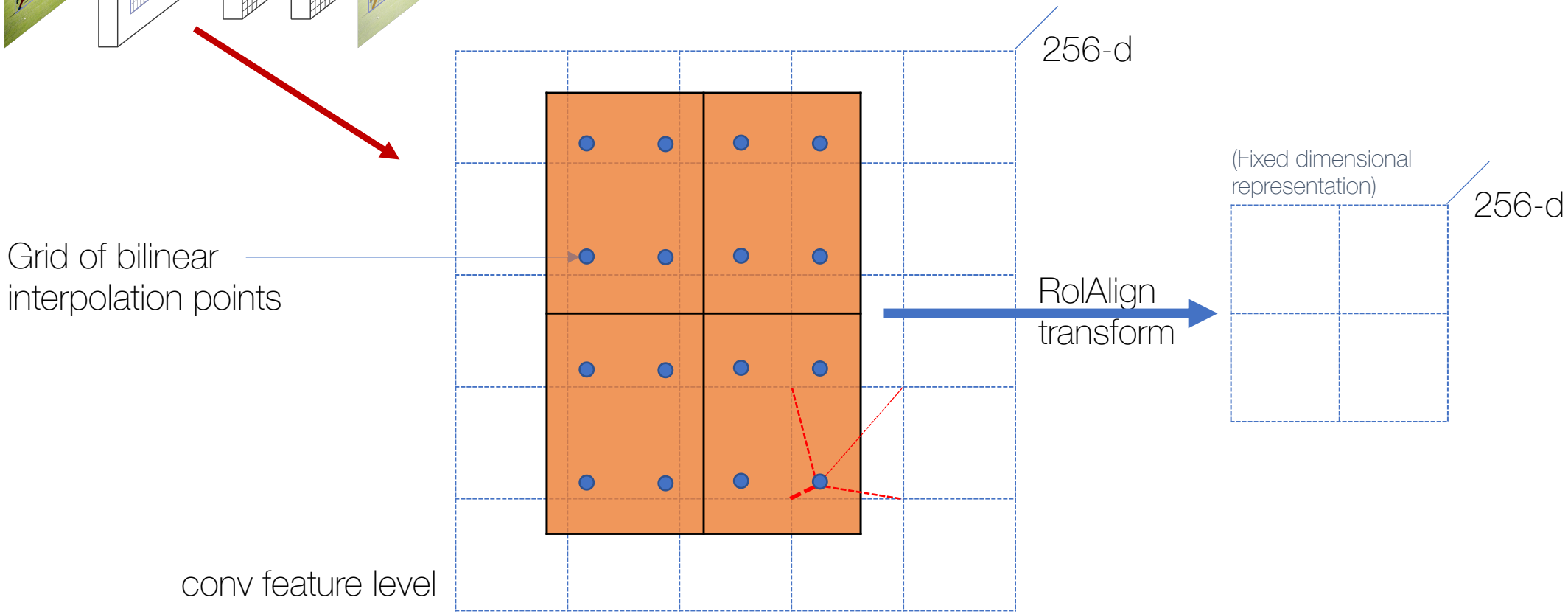
RoIAlign vs RoIPool

- RoIPool *breaks* pixel-to-pixel translation-equivariance



RoIAlign


RoIAlign
class
box
conv
conv


256-d
Grid of bilinear
interpolation points
RoIAlign
transform
(Fixed dimensional
representation)
256-d
conv feature level

Mask R-CNN for Instance Segmentation

Mask R-CNN for Pose Prediction

Task

Given an image:

- Detect all human instances in the image
- Localize their landmarks, e.g. Right Shoulder, Nose etc.

Mask R-CNN for Pose Prediction

Loss

- Assume a set of K landmarks, e.g. Nose, Right Shoulder etc.
- For a detected instance R
  - corresponding logits J predicted by the Pose Head of Mask R-CNN of shape H x W x K
  - Ground truth locations $J_{gt} = \{(x_k, y_k)$, for k = 0, …, K-1$\}$

Content

- A Brief Intro
- Visual Recognition: Problem Definitions & Tasks
  - Semantic Segmentation
    - FCN
  - Object Detection
    - Fast(er) R-CNN
  - Instance Segmentation & Pose Prediction
    - Mask R-CNN
- ImageNet in 1H: Important Lessons
- Useful Tips

Making It Work

Solving a problem:

- Modeling: Transform the problem into a computational algorithm
- Architecture: Design the hierarchical model to capture the information necessary
- Invariance & Equivariance Properties
- Efficiency

Making It Work

Solving a problem:

- Modeling: Transform the problem into a computational algorithm
- Architecture: Design the hierarchical model to capture the information necessary
- Invariance & Equivariance Properties
- Efficiency
- **Hyperparameter Tuning**

Making It Work

Solving a problem:

- Modeling: Transform the problem into a computational algorithm
- Architecture: Design the hierarchical model to capture the information necessary
- Invariance & Equivariance Properties
- Efficiency
- **Hyperparameter Tuning: batch size, learning rate, epochs, decay, weight loss etc.**

Hyperparameters & SGD

- Batch Size: #gpus * #batch_per_gpu
- Learning Rate
- Training Schedule
- Weight Loss (important for multi-task learning)

Hyperparameters & SGD

- Batch Size: #gpus * #batch_per_gpu
- Learning Rate
- Training Schedule
- Weight Loss (important for multi-task learning)

These hyperparameters are correlated

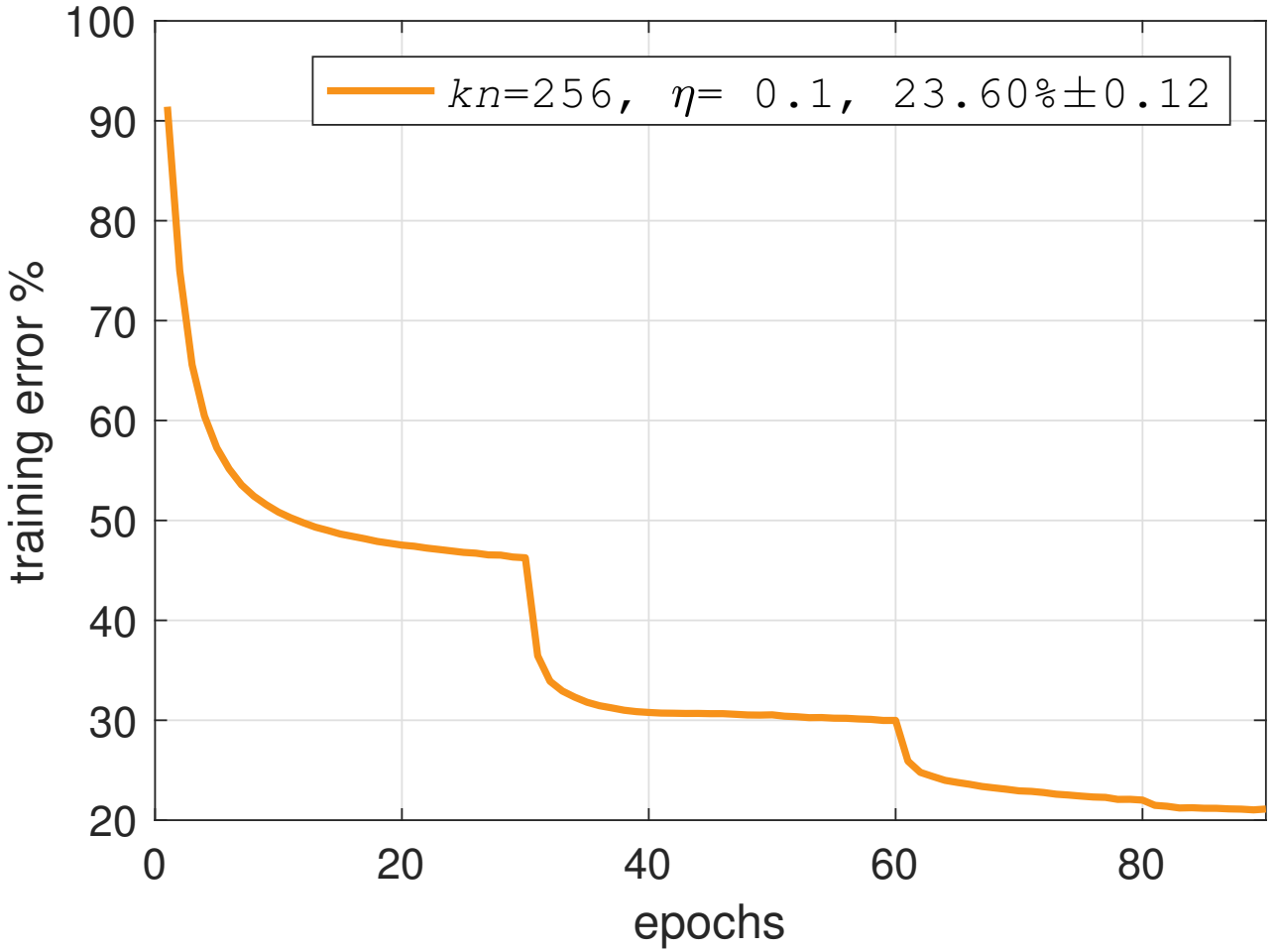
ImageNet Training

Original Training Schedule [1]:

```
BASE_LR: 0.1
STEP_SIZES: [150150, 150150, 150150] # 30, 30, 30 epochs
LRS: [1, 0.1, 0.01, 0.001]
MAX_ITER: 500500 # total 100 epochs
```

[1] He et al, Deep Residual Learning for Image Recognition, CVPR 2016

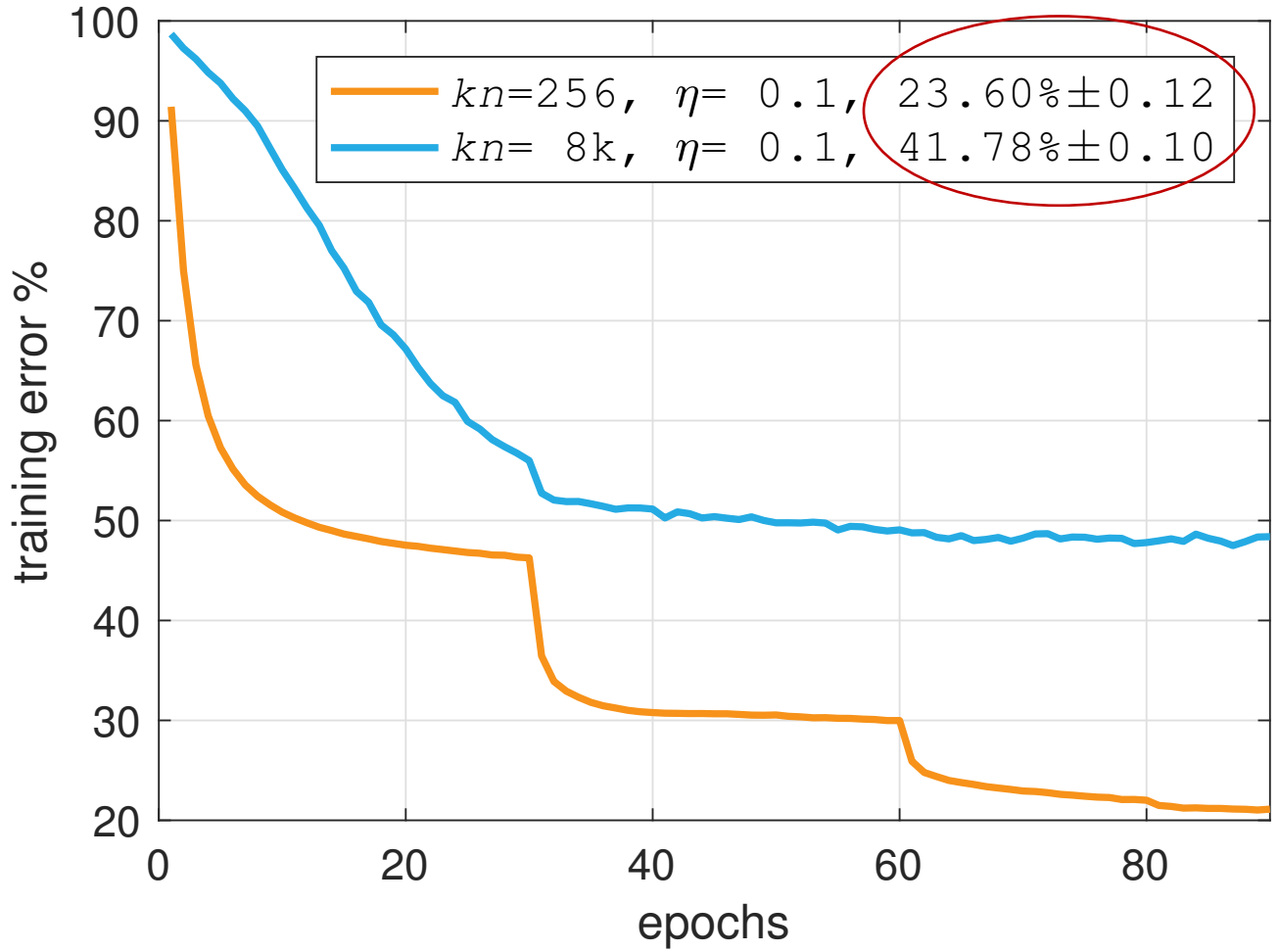
ImageNet Training



- k = #gpus
- n = per gpus batch size
- η = learning rate
- 256 = 8 x 32

[2] Goyal et al, Accurate, Large Minibatch SGD: Training ImageNet in 1 Hour, 2017

ImageNet in 1H [2]



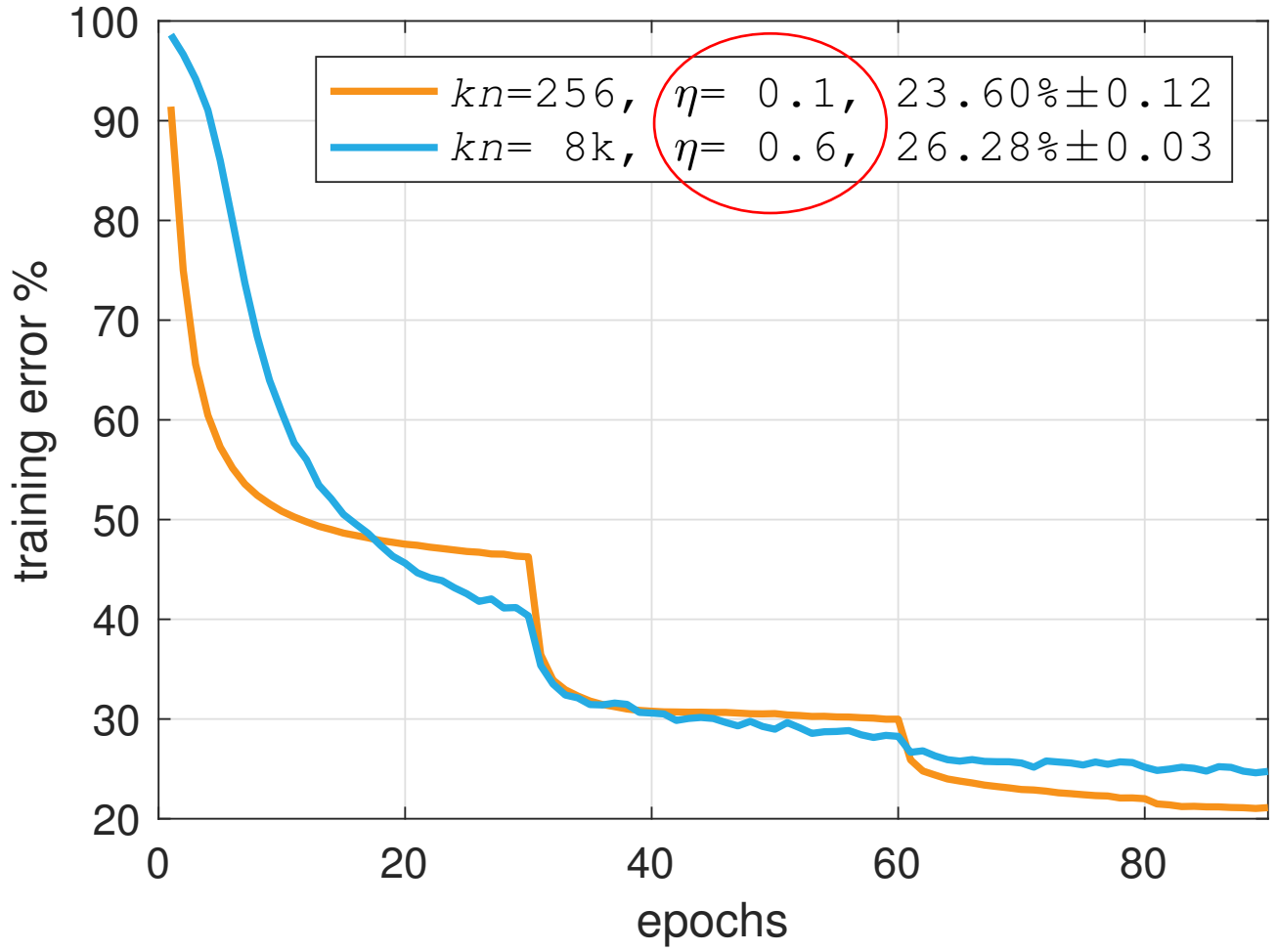
8192 = 256 x 32

#gpus

per gpu batch

[2] Goyal et al, Accurate, Large Minibatch SGD: Training ImageNet in 1 Hour, 2017

ImageNet in 1H [2]

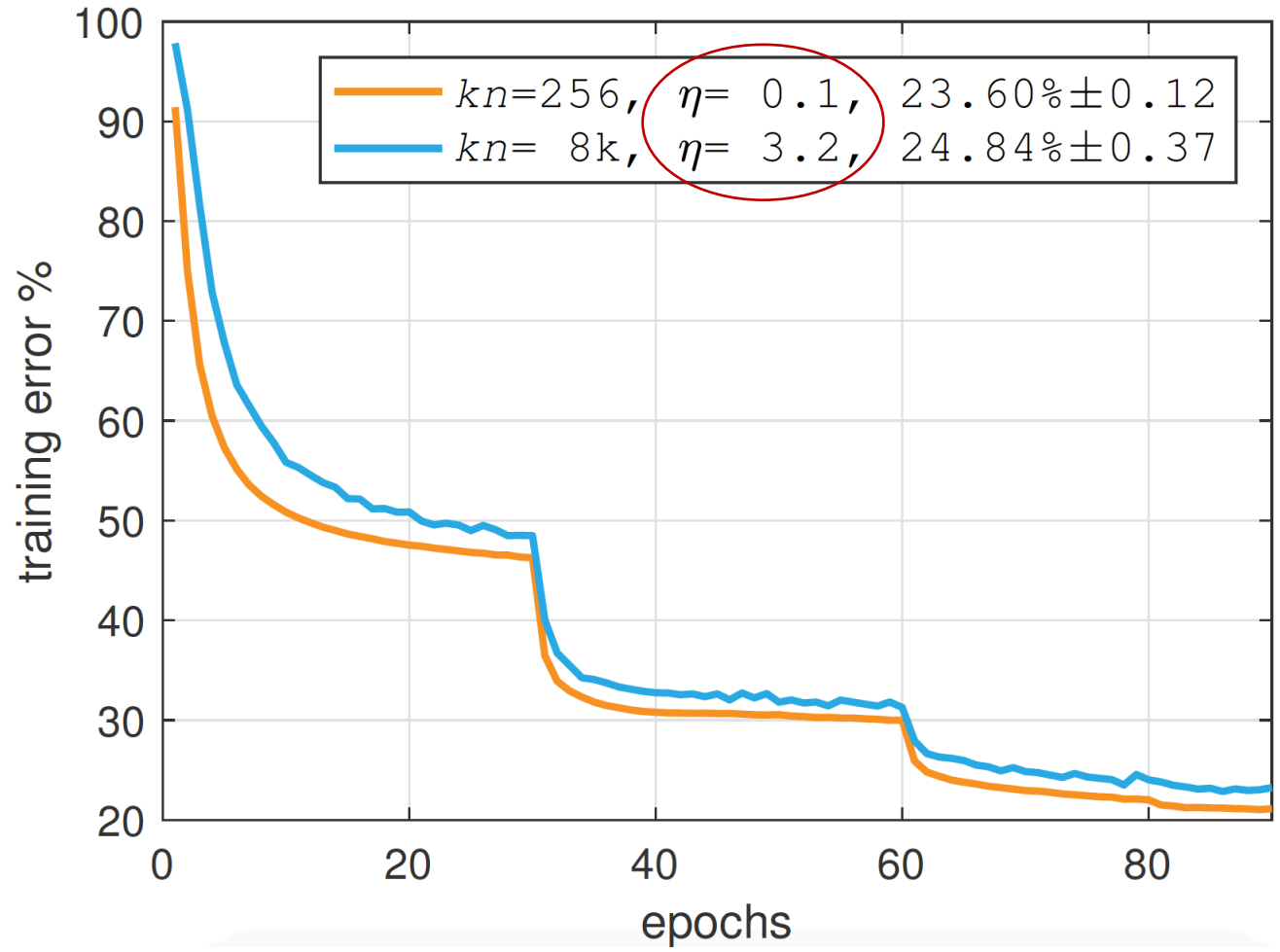


Sqrt LR scaling

$$\eta \leftarrow \sqrt{(\frac{8192}{256})} \cdot \eta$$

[2] Goyal et al, Accurate, Large Minibatch SGD: Training ImageNet in 1 Hour, 2017

ImageNet in 1H [2]

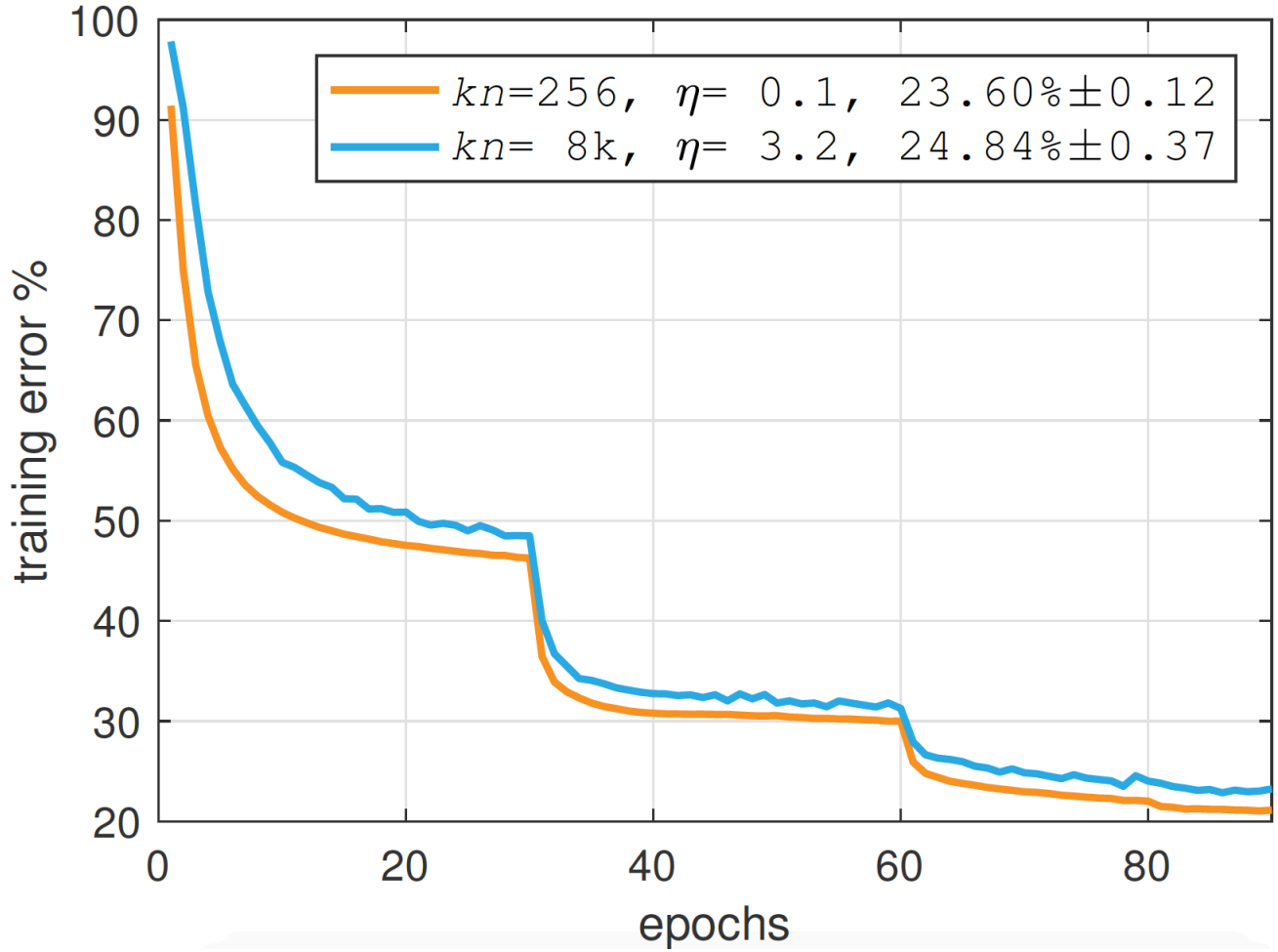


Linear scaling

$$\eta \leftarrow \frac{8192}{256} \cdot \eta$$

[2] Goyal et al, Accurate, Large Minibatch SGD: Training ImageNet in 1 Hour, 2017

ImageNet in 1H [2]



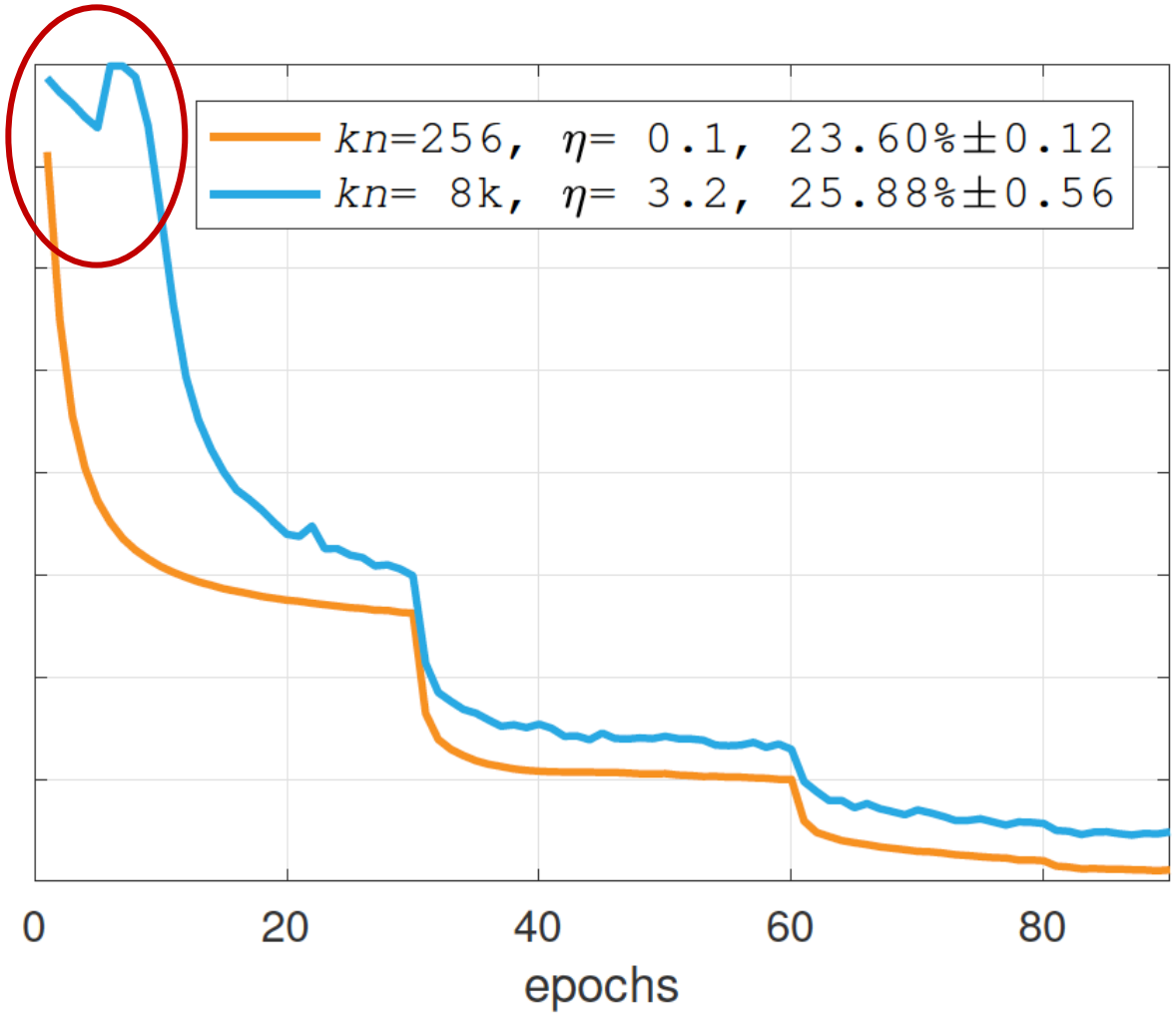
Linear scaling

$$\eta \leftarrow \frac{8192}{256} \cdot \eta$$

Optimization difficulty

[2] Goyal et al, Accurate, Large Minibatch SGD: Training ImageNet in 1 Hour, 2017

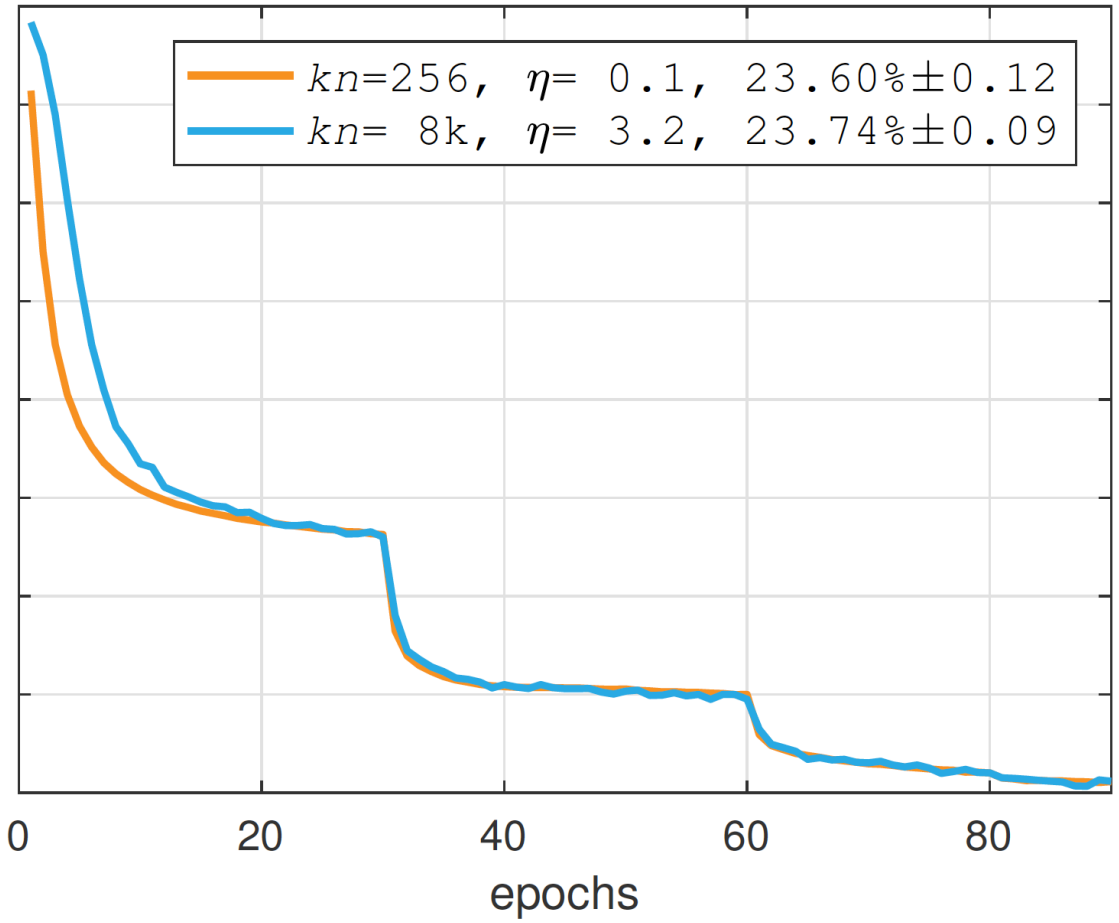
ImageNet in 1H [2]



Rapid changes in network in the beginning of training → use small LR for first few epochs (**constant LR warmup**)

[2] Goyal et al, Accurate, Large Minibatch SGD: Training ImageNet in 1 Hour, 2017

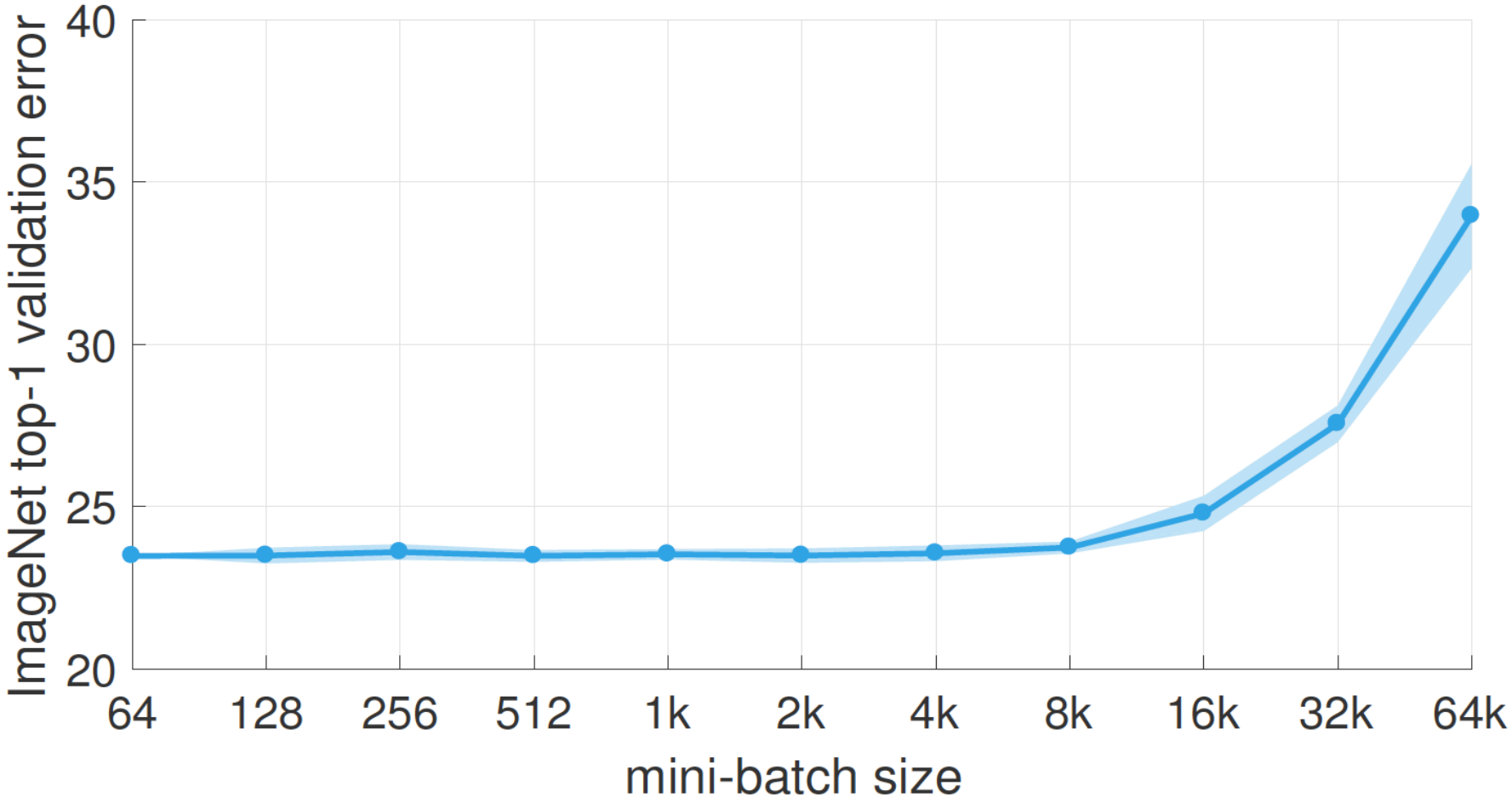
ImageNet in 1H [2]



start from LR of η(=0.1) and increase it by constant amount (0.003) at each iteration so that $\hat{\eta} = k\eta$ $(= 3.2)$ after 5 epochs (**gradual LR warmup**)

[2] Goyal et al, Accurate, Large Minibatch SGD: Training ImageNet in 1 Hour, 2017

ImageNet in 1H [2]



[2] Goyal et al, Accurate, Large Minibatch SGD: Training ImageNet in 1 Hour, 2017

Content

- A Brief Intro
- Visual Recognition: Problem Definitions & Tasks
  - Semantic Segmentation
    - FCN
  - Object Detection
    - Fast(er) R-CNN
  - Instance Segmentation & Pose Prediction
    - Mask R-CNN
- ImageNet in 1H: Important Lessons
- Useful Tips

Common Mistakes

- Remember that if you reduce the effective batch size you need to reduce the learning rate

Common Mistakes

- Remember that if you reduce the effective batch size you need to reduce the learning rate
- When using #gpus >1 be aware of how the gradients are combined (added vs. averaged) → tune learning rate and loss weight accordingly

```
kps_prob, loss_kps = model.net.SoftmaxWithLoss(
    ['kps_score_reshaped', 'keypoint_locations_int32', 'keypoint_weights'],
    ['kps_prob', 'loss_kps'],
    scale=cfg.KRCNN.LOSS_WEIGHT / cfg.NUM_GPUS,
    spatial=0
)
```

Common Mistakes

- Remember that if you reduce the effective batch size you need to reduce the learning rate
- When using #gpus >1 be aware of how the gradients are combined (added vs. averaged) → tune learning rate and loss weight accordingly
- Count training duration in epochs not in training iterations: two models are comparable when they have been trained for the same number of epochs (this might not be the same number of iterations)

How to do research

- Read old & new papers
- Identify what you are passionate about
- Start from the basics and build up (instead of the opposite!)
- Ablate all the parts of your approach
- Visualize your input, output, intermediate representations
- Do not use deep learning as a black box
  - Understand the **forward** flow of computation
  - Understand the **backward** flow of computation

  (see: Andrej's blog post)